Multiple Cue Integration for Robust Tracking in Dynamic Environments: Application to Video Relighting

By

Francesc Moreno Noguer

Institut de Robòtica i Informàtica Industrial

Universitat Politècnica de Catalunya

Consejo Superior de Investigaciones Científicas

A thesis co-directed by:

Alberto Sanfeliu

Peter N. Belhumeur

A thesis submitted for the degree of

Doctor of Philosophy

Barcelona, September 2005

Universitat Politècnica de Catalunya

Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial

PhD program:
Control, Visió i Robòtica

This thesis was completed at:

Institut de Robòtica i Informàtica Industrial, UPC-CSIC

Thesis advisors:
Alberto Sanfeliu
Peter N. Belhumeur

PhD Committee:
Pere Brunet, President
Alícia Casals, Secretary
Joan Martí
Dimitris Samaras
Jan-Olof Eklundh

To Ramon, Irene, Fernando and Victòria

Acknowledgements

First, I would like to express sincere thanks to Professor Alberto Sanfeliu, who introduced me into the computer vision field some years ago, in the 'Robotics and Vision' subject during my bachelor, in 1997, and later become the advisor of the present dissertation. To Peter Belhumeur and Shree Nayar, for offering me an exciting project to work on during my visit at Columbia University. Also to Dimitris Samaras for his wise advice, and fruitful comments and discussions at fancy New York bars.

I would like to thank also to Rafael, Carme, Jordi and Fede for their invaluable support throughout these years at IRI, and the rest of the people from 'room 19': Juan, Guillem, Tere, Josep Marias, José Luís and Andreu, for sharing with me so much experiences (in so little space), and bearing with no complaint the high volume of my headphones. I am grateful also to Anne, Anuj, Nigel, Jaime, Tolga, Ko, Kalyan, Dhruv, Kshitiz, Sujit and Assaf, for making my stay at Columbia a delightful experience.

Financial support was provided by a fellowship from the Spanish Ministry of Science and Technology and CICYT projects DPI2004-05414, DPI2001-2223, and DPI2000-1352-C02-01 (thanks to Ramon López de Mántaras and Enric Celaya). Also, part of my stay at Columbia University was sponsored by the National Science Foundation Award No. IIS-03-08185 'Complex Reflectance, Texture and Shape: Methods and Representations for Object Modeling'.

I finalment, i més especialment, estic enormement agraït als meus pares Fernando i Victòria i al meu germà Ramon, pel suport incondicional que m'han donat al llarg de tots aquests anys. I a la meva estimada Irene, gràcies per estar al meu costat. Tot el recolzament diari que m'heu donat, fa que aquesta tesi us pertanyi tant a vosaltres com a mi.

Abstract

Motion analysis and object tracking has been one of the principal focus of attention over the past two decades within the computer vision community. The interest of this research area lies in its wide range of applicability, extending from autonomous vehicle and robot navigation tasks to entertainment and virtual reality applications.

Even though impressive results have been obtained in specific problems, object tracking is still an open problem, since available methods are prone to be sensitive to several artifacts and non-stationary environment conditions, such as unpredictable target movements, gradual or abrupt changes of illumination, similar objects proximity or cluttered backgrounds. Multiple cue integration has been proved to enhance the robustness of the tracking algorithms in front of such disturbances. In recent years, due to the increasing power of the computers, there has been a significant interest in building complex tracking systems which simultaneously consider multiple cues. However, most of these algorithms are based on heuristics and *ad-hoc* rules formulated for specific applications, making impossible to extrapolate them to new environment conditions.

In this dissertation we propose a general probabilistic framework to integrate as many object features as necessary, permitting them to mutually interact in order to obtain a precise estimation of its state, and thus, a precise estimate of the target position. This framework is utilized to design a tracking algorithm, which is validated on several video sequences involving abrupt position and illumination changes, target camouflaging and non-rigid deformations. Among the utilized features to represent the target, it is important to point out the use of a robust parameterization of the target color in an object dependent colorspace which allows to distinguish the object from the background more clearly than other colorspaces commonly used in literature.

In the last part of the dissertation, we design an approach to relighting static and moving scenes with unknown geometry. The relighting is performed by an 'image-based' methodology, in which the rendering under new lighting conditions is achieved by linear combinations of a set of pre-acquired reference images of the scene illuminated by known light patterns. Since the placement and brightness of the light sources composing such light patterns can be controlled, it is natural to ask: what is the optimal way to illuminate the scene to reduce the number of reference images that are needed? We show that the best way to light the scene (i.e., the way that minimizes the number of reference images) is not using a sequence of single, compact light sources as is most commonly done, but rather to use a sequence of lighting patterns as given by an *object-dependent* lighting basis. It is important to note that when relighting video sequences, consecutive images need to be aligned with respect to a common coordinate frame. However, since each frame is generated by a different light pattern illuminating the scene, abrupt illumination changes are produced between consecutive reference images. Under these circumstances, the tracking framework designed in this dissertation plays a central role. Finally, we present several relighting results on real video sequences of moving objects, moving faces, and scenes containing both. In each case, although a single video clip was captured, we are able to relight again and again, controlling the lighting direction, extent, and color.

Resum

L'anàlisi de moviment i seguiment d'objectes ha estat un dels pricipals focus d'atenció en la comunitat de visió per computador durant les dues darreres dècades. L'interès per aquesta àrea de recerca resideix en el seu ample ventall d'aplicabilitat, que s'extén des de tasques de navegació de vehicles autònoms i robots, fins a aplicacions en la indústria de l'entreteniment i realitat virtual.

Tot i que s'han aconseguit resultats espectaculars en problemes específics, el seguiment d'objectes continua essent un problema obert, ja que els mètodes disponibles són propensos a ser sensibles a diversos factors i condicions no estacionàries de l'entorn, com ara moviments impredictibles de l'objecte a seguir, canvis suaus o abruptes de la il·luminació, proximitat d'objectes similars o fons confusos. Enfront aquests factors de confusió la integració de múltiples característiques ha demostrat que permet millorar la robustesa dels algoritmes de seguiment. En els darrers anys, degut a la creixent capacitat de càlcul dels ordinadors, hi ha hagut un significatiu increment en el disseny de complexes sistemes de seguiment que consideren simultàniament múltiples característiques de l'objecte. No obstant, la majoria d'aquests algoritmes estan basats en heurístiques i regles *ad-hoc* formulades per aplicacions específiques, fent-ne impossible l'extrapolació a noves condicions de l'entorn.

En aquesta tesi proposem un marc probabilístic general per integrar el nombre de característiques de l'objecte que siguin necessàries, permetent que interactuin mútuament per tal d'estimar-ne el seu estat amb precisió, i per tant, estimar amb precisió la posició de l'objecte que s'està seguint. Aquest marc, s'utilitza posteriorment per dissenyar un algoritme de seguiment, que es valida en diverses seqüències de vídeo que contenen canvis abruptes de posició i il·luminació, camuflament de l'objecte i deformacions no rígides. Entre les característiques que s'han utilitzat per representar l'objecte, cal destacar la parametrització robusta del

color en un espai de color dependent de l'objecte, que permet distingir-lo del fons més clarament que altres espais de color típicament ulitzats al llarg de la literatura.

En la darrera part de la tesi dissenyem una tècnica per re-il·luminar tant escenes estàtiques com en moviment, de les que s'en desconeix la geometria. La re-il·luminació es realitza amb un mètode 'basat en imatges', on la generació de les images de l'escena sota noves condicions d'il·luminació s'aconsegueix a partir de combinacions lineals d'un conjunt d'imatges de referència pre-capturades, i que han estat generades il·luminant l'escena amb patrons de llum coneguts. Com que la posició i intensitat de les fonts d'il·luminació que formen aquests patrons de llum es pot controlar, és natural preguntar-nos: quina és la manera més òptima d'il·luminar una escena per tal de reduir el nombre d'imatges de referència? Demostrem que la millor manera d'il·luminar l'escena (és a dir, la que minimitza el nombre d'imatges de referència) no és utilitzant una seqüència de fonts d'il·luminació puntuals, com es fa generalment, sinó a través d'una seqüència de patrons de llum d'una base d'il·luminació dependent de l'objecte. És important destacar que quan es re-il·luminen seqüències de vídeo, les imatges successives s'han d'alinear respecte a un sistema de coordenades comú. Com que cada imatge ha estat generada per un patró de llum diferent il·luminant l'escena, es produiran canvis d'il·luminació bruscos entre imatges de referència consecutives. Sota aquestes circumstàncies, el mètode de seguiment proposat en aquesta tesi juga un paper fonamental. Finalment, presentem diversos resultats on re-il·luminem seqüències de vídeo reals d'objectes i cares d'actors en moviment. En cada cas, tot i que s'adquireix un únic vídeo, som capaços de re-il·luminar una i altra vegada, controlant la direcció de la llum, la seva intensitat, i el color.

Notation

Bayesian tracking

$\mathbf{x}$	state vector
$\mathbf{z}$	measurement vector
$\mathbf{Z}$	matrix of measurement vectors
$p(\mathbf{x})$	prior density
$p(\mathbf{x}\|\mathbf{Z})$	posterior density
$p(\mathbf{x}^t\|\mathbf{x}^{t-1})$	dynamic model
$p(\mathbf{z}\|\mathbf{x})$	observation (or measurement) model

Kalman filter

$\mathbf{\Sigma}$	state covariance
$\mathbf{D}$	process linear dynamic model
$\mathbf{q}_d$	process noise
$\mathbf{\Sigma}_d$	process covariance noise
$\mathbf{M}$	observation (or measurement) linear dynamic model
$\mathbf{q}_m$	observation noise
$\mathbf{\Sigma}_m$	observation covariance noise
$\mathbf{K}$	Kalman gain
$\mathbf{x}_-$	a priori state estimate
$\mathbf{x}_+$	a posteriori state estimate
$\mathbf{\Sigma}_-$	a priori state covariance estimate
$\mathbf{\Sigma}_+$	a posteriori state covariance estimate
$\mathbb{I}$	identity matrix
$\mathbf{0}$	null matrix

Particle filters

$\mathbf{s}_i$	i-th state vector sample
π_i	weight associated to i-th state vector sample
n	number of particles
$\sim$	sampling with replacement operation
$\otimes$	convolution operation between probability density functions
$\times$	multiplication operation between probability density functions

Feature extraction

$\mathbf{c} \in \mathbb{R}_{3\times1}$	pixel color represented in the RGB colorspace
$\mathbf{C}$	matrix of pixels represented in the RGB colorspace
$\mathbf{f} \in \mathbb{R}_{2\times1}$	pixel color represented in the Fisher colorspace
$\mathbf{F}$	matrix of pixels represented in the Fisher colorspace
$\mathcal{O}$	object or foreground region of the image
$\mathcal{B}$	background region of the image
$\varepsilon = \{\mathcal{O}, \mathcal{B}\}$	foreground-background class index
$\mathbf{S}_w$	within class scatter matrix
$\mathbf{W} \in \mathbb{R}_{2\times3}$	Fisher plane ($\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2]^T$)
$\mathbf{S}_w$	within class scatter matrix
$\mathbf{S}_b$	between class scatter matrix
$\mathbf{\Sigma}_b$	non-parametric between class scatter matrix
m_ε	number of Gaussian components fitted to the ε-class color distribution
$\boldsymbol{\mu}_{\varepsilon,j}$	mean of the j-th Gaussian component fitted to the ε-class color distribution
$\mathbf{\Sigma}_{\varepsilon,j}$	covariance of the j-th Gaussian component fitted to the ε-class color distribution
$\mathbf{u}$	pixel position in image ($\mathbf{u} = [u, v]^T$)
$\mathbf{R}$	matrix of contour points
$\mathbf{Q}$	matrix containing the snake rigidity and elasticity parameters

$\mathbf{H}$	affine snake dynamic model
$\mathcal{W}$	object bounding box

Multiple cue integration

$\mathbf{x}_i$	state vector associated to i-th object feature
$\mathbf{X}$	complete state vector representing the target ($\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$)
$E(\mathbf{x}_i)$	expected value of the i-th feature state
$\mathcal{D}$	survival diagnostic
n_i	number of samples approximating the $\mathbf{x}_i$ state vector
$\mathbf{s}_{ij}$	j-th sample approximating the state of i-th feature
π_{ij}	weight associated to sample $\mathbf{s}_{ij}$
$\mathbf{H}_i$	scaling factor in the dynamic model of feature $\mathbf{x}_i$
$\mathbf{q}_i$	translational factor in the dynamic model of feature $\mathbf{x}_i$

Video relighting

$\mathbf{i}$	gray level or single band image
$\mathbf{I}$	multiband image or array containing multiple single band images
$\mathbf{l}_p$	p-th light pattern
$L_p(\boldsymbol{\Phi_l})$	radiance of the l-th light source in p-th light pattern
$\boldsymbol{\Phi}_l$	global spherical coordinates of the l-th light source ($\boldsymbol{\Phi}_l = [\theta_l, \phi_l]^T$)
$\mathbf{L}$	matrix of light patterns
$\mathbf{E}$	matrix of single light source patterns
$R_{\mathbf{u}_i}(\boldsymbol{\Phi}_l)$	reflectance of pixel $\mathbf{u}_i$ as a result of illumination from direction $\boldsymbol{\Phi}_l$
$\mathbf{R}$	reflectance matrix (contains the reflectances $R_{\mathbf{u}_i}(\boldsymbol{\Phi}_l)$ $\forall$ pixel $\mathbf{u}_i$ and $\forall$ light direction $\boldsymbol{\Phi}_l$)
$\mathbf{D}$	decoding matrix
$\Upsilon(\cdot)$	geometric and appearance warping function
$\|\cdot\|$	Frobenius or Euclidean norm
$\lvert\cdot\rvert$	absolute value function

Nomenclature

PDF	Probability Density Function
EM	Expectation Maximization
LDA	Linear Discriminant Analysis
KNN	K-Nearest Neighbours
GVF	Gradient Vector Flow
MoG	Mixture of Gaussians
$\mathcal{BF}$	Bayesian Filter
$\mathcal{KF}$	Kalman Filter
$\mathcal{PF}$	Particle Filter

Contents

List of Figures

Chapter 1

Introduction

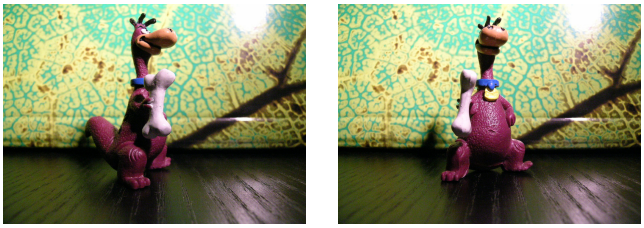
Nowadays, digital video cameras are becoming daily devices in our lives. Their relatively low cost and portability properties have made them a popular travel mate for almost everybody. Furthermore, apart from the conventional handycams of personal use, we can find video cameras in a high variety of places and situations, for instance, mounted in building entrances, in streets and highways, integrated into other instruments such as cell phones or pens, and we can also find the so-called wearable video cameras, which are usually mounted on glasses or caps and may be 'worn' by the user. This development has grown in parallel with the capacity of the data storage devices, and as a consequence, large amounts of visual data may be collected by the video cameras. Computer vision technology offers the possibility of automatically or semi-automatically process these data, and perform tasks with a certain level of 'intelligence'.

In most of the tasks involving the processing of video sequences, motion analysis and object tracking play a central role. Actually, this specific area of the computer vision has been one the the principal focus of attention of the research over the past two decades, with increasingly impressive results in diverse tasks, extending from the autonomous vehicle and robot navigation to entertainment and virtual reality applications.

Nevertheless, in spite of this variety of results and applications, object tracking is still an open problem, since available methods are prone to be sensitive to several artifacts and non-stationary conditions. We can enumerate a list of them:

- **Changes of object pose and scale** caused by the relative movement of the object with respect to the camera. Furthermore, these movements might be abrupt.
- **Non-rigid object deformations** produced by some collisions or external forces applied to the target.

Illumination disturbances



Unexpected target movements

Occlusions and cluttered backgrounds

Figure 1.1: **Illustrative examples of the disturbances that need to be addressed in a tracking application.** First row: Illumination disturbances, caused by movements of the light source and cast shadows. Second row: Unexpected target movements. Third row: occlusions of the target and camouflage caused by cluttered backgrounds.

- **Gradual illumination changes**, usually generated by a smooth movement of the object with respect to the light source, or conversely, produced by a smooth movement of the light source (for instance the daylight generates gradual illumination changes).
- **Abrupt illumination changes** are one of the most critical difficulties to deal with, and use to be caused by light switching or sudden occlusions of the light source.
- **Complex surface properties**, in terms of geometry and reflectance, might cause complex illumination effects such as specularities or intereflections.

- **Shadows**, especially the cast shadows produced when the target casts on itself or different close objects cast shadows on the target.

- **Camouflage** of the tracked object due to cluttered backgrounds or backgrounds having a similar appearance to that of the target.

The target representation is an important initial issue that needs to be addressed when designing any visual tracking system. In the literature, the visual modules most commonly used in tracking tasks include geometric cues, motion, color, contrast, textures, appearance and shape. Distinct object features show different behaviours under the previously mentioned disturbances. For instance, color usually is a robust cue under non-rigid deformations of the object, and by contrast, it is sensitive to gradual illumination changes. Conversely, the contour cue tends to be susceptible to non-rigid deformations and robust to gradual illumination changes.

If the target moves in laboratory conditions under controlled lighting conditions, the effect of the disturbances previously mentioned may be reduced, and the representation of the target by a single feature might suffice. However, when the object moves out of these controlled conditions, the representation of the object by a single feature is no longer effective. In these circumstances the use of simultaneous redundant or complementary cues might significantly enhance the performance of the tracking.

In recent years, due to the increasing power of the computers, there has been a significant interest in building complex tracking systems which simultaneously consider multiple cues. However, as it will be seen in Chapter 4, most of these algorithms are based on heuristics and ad-hoc rules formulated for specific applications.

The goal of this dissertation is to establish a new and general probabilistic framework for tracking purposes, allowing to integrate as many features as necessary, without increasing dramatically the complexity of the system. As it will be seen in Chapter 5, the proposed methodology improves the performance of existing approaches. Furthermore, in Chapter 3 we focus on the feature selection stage, and in particular, we contribute with a new color space, which we prove, is appropriate for tracking tasks. Finally, in Chapter 6, the proposed tracking method is utilized to develop a video relighting methodology, which requires a tracking system robust to abrupt illumination changes. This chapter also contributes a detailed study about the optimal way to reilluminate a video sequence.

In the rest of this chapter, we briefly discuss on the main contributions, give an outline of the rest of the dissertation, and include a reference of the related publications.

1.1 Main contributions

The main contributions of the dissertation may be summarized as follows:

1. We propose a probabilistic framework to integrate as many features as necessary, for tracking tasks. This method allows to integrate both conditionally dependent and conditionally independent object cues, and any feature whose state vector is represented by a *Probability Density Function* (PDF). In addition, the complexity of the system does not increase dramatically with the number of integrated cues.

2. The proposed framework is applied to develop a robust tracking system that simultaneously accommodates both geometric and appearance object features. With this approach we can deal with challenging video sequences suffering from cluttered backgrounds, unexpected object dynamics, non-rigid deformations and abrupt illumination changes.

3. The specific representation and estimation procedure proposed for the color feature are also novel contributions of this dissertation: the color feature is estimated through a particle filter formulation, and represented on an adaptable colorspace dependent on the tracked object.

4. The tracking algorithm is utilized to develop a video relighting algorithm, which requires from the alignment of consecutive images suffering from abrupt illumination changes. Apart from the application itself (video relighting is a novel field with only a small number of previous contributions), we contribute by studying which is the optimal way of illuminating a video sequence of a moving scenario, in order to subsequently relight it with the minimum cost.

1.2 Thesis overview

The dissertation is organized according to the following chapters:

- **Chapter 2** introduces the foundations of the visual tracking problem, seen as a Bayesian process where conditional densities are propagated. Two Bayesian filters are analyzed

from this point of view, namely, the *Kalman filter* and *particle filters*, emphasizing that their operation may be described through the same stages: hypotheses generation (or prediction) and hypotheses correction. Precisely, these filters will be used in order to estimate the state of individual object features, and integrated in the probabilistic framework proposed in Chapter 5.

- **Chapter 3** describes the features that will be used to robustly represent the target, including appearance and geometric cues. In this chapter, special attention will be given to the selection of the colorspace where image points are represented. An object dependent colorspace is proposed (we call it *Fisher colorspace*), which has the capacity to maximize the distance between the color representation of the target points from the color representation of the background points. Note that this special feature is desirable for tracking tasks.

  Furthermore, the rest of object features will be described. In particular, we describe the parameterization of the color points on the Fisher colorpace, through a *Mixture of Gaussians* (MoG) model, the representation of the contour by a snake formulation, and a rough estimate of the object position by a rectangular bounding box.

- **Chapter 4** reviews previous approaches to integration of visual modules for tracking and figure/background segmentation tasks. Based on the classification that Clark and Yuille (23) suggested to classify general sensor fusion techniques, two major categories are distinguished, namely the weakly coupled and strongly coupled, depending on the degree of interaction between the visual modules. The taxonomy is completed and made more precise by considering several subcategories. Over 50 papers (including the most relevant works on the field over the last decade) are reviewed and classified into the proposed taxonomy. Finally, the reviewed papers are analyzed in terms of robustness, adaptability and complexity, which are properties that need to be considered when designing a tracking system integrating various features.

- **Chapter 5** focuses on the proposed framework for multiple cue integration for robust tracking. The analysis and the techniques described in previous chapters are gathered in order to propose a general probabilistic framework, allowing to integrate any number of features. Initially, the method is theoretically described and validated by a simple synthetic example which is used as a benchmark to compare the performance of the

method suggested in the dissertation, to that of other related approaches. In the second part of the chapter, based on the model just defined, we design a tracking algorithm able to cope with several artifacts and non-linearities. Various tracking examples in diverse and dynamic environments are presented.

- **Chapter 6** describes the development of a video relighting application, where the tracking methodology proposed in Chapter 5 plays an important role for aligning consecutive images that suffer from abrupt illumination changes. This chapter also analyzes the video relighting problem in detail, and proposes the use of an object dependent lighting basis to illuminate the object, which minimizes the relighting cost. Both synthetic and real experiments show that the proposed lighting basis outperforms other existent bases.

- **Chapter 7** summarizes the dissertation, and sums up the contributions. Future research directions are also discussed.

1.3 Derived publications

The following is a list of the published work derived from this thesis:

1. F.Moreno-Noguer, A.Sanfeliu, D.Samaras, "Integration of deformable contours and a multiple hypotheses Fisher color model for robust tracking in varying illuminant environments", *'Minor changes' after first correction stage in Image and Vision Computing*.

2. F.Moreno-Noguer, S.K.Nayar, P.N.Belhumeur, "Optimal Illumination for Image and Video Relighting (full paper)", *IEE European Conference on Visual Media Production (CVMP)*, 2005.

3. F.Moreno-Noguer, A.Sanfeliu, D.Samaras, "Integration of Conditionally Dependent Object Features for Robust Figure/Background Segmentation", *Proc. IEEE International Conference on Computer Vision (ICCV)* , 2005.

4. F.Moreno-Noguer, S.K.Nayar, P.N.Belhumeur, "Optimal Illumination for Image and Video Relighting (short sketch)", *SIGGRAPH Sketch*, 2005.

5. F.Moreno-Noguer, A.Sanfeliu, "A Framework to Integrate Particle Filters for Robust Tracking in Non-stationary Environments", *Proc. Iberian Conference on Pattern Recognition and Image Analysis (IBPRIA), LNCS 3522*, 2005 , **BEST PAPER AWARD**.

6. F.Moreno-Noguer, A.Sanfeliu, "Integration of Shape and a Multihypotheses Fisher Color Model for Figure-Ground Segmentation in Non-Stationary Environments", *Proc. International Conference on Pattern Recognition (ICPR)*, Vol.4, pp.771-774, 2004.

7. F.Moreno-Noguer, A.Sanfeliu, D.Samaras, "Fusion of a Multiple Hypotheses Color Model and Deformable Contours for Figure Ground Segmentation in Dynamic Enviroments", *Proc. Workshop on Articulated and Non-Rigid Motion (in conjunction with CVPR'04)*, 2004.

8. F.Moreno-Noguer, A.Sanfeliu, "Adaptative Color Model for Figure Ground Segmentation in Dynamic Environments", *Proc. Iberoamerican Congress on Pattern Recognition (CIARP)*, pp.37-44, 2004.

9. F.Moreno-Noguer, J.Andrade-Cetto, A.Sanfeliu, "Fusion of Color and Shape for Object Tracking under Varying Illumination", *Proc. Iberian Conference on Pattern Recognition and Image Analysis (IBPRIA), LNCS 2652*, pp.580-588, 2003.

10. F.Moreno-Noguer, A.Tarrida, J.Andrade-Cetto, A.Sanfeliu, "3D Real Time Head Tracking Fusing Color Histograms and Stereovision", *Proc. International Conference on Pattern Recognition (ICPR)*, Vol.1, pp.368-371, 2002.

11. F.Moreno-Noguer, "Pattern recognition systems", *Final Year Project for the Electrical Engineering Degree, University of Barcelona*, 2002.

12. F.Moreno-Noguer, J.Andrade-Cetto, A.Sanfeliu, "Localization of Human Faces Fusing Color Segmentation and Depth from Stereo", *Proc. IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp.527-536, 2001.

13. F.Moreno-Noguer, "Development of a stereo vision system for a mobile robot", *Final Year Project for the Industrial Engineering Degree, Technical University of Catalonia*, 2001.

Chapter 2

Tracking viewed as a temporal propagation of conditional densities

In this chapter we will establish the bases of the tracking problem and introduce the formulation that will be used in the following chapters. The tracking will be explained as a hypotheses generation and correction process in a Bayesian framework, which is equivalent to a propagation of conditional densities through the time, where the conditional densities are represented by *Probability Density Functions* (PDF's). Two methodologies will be briefly described, namely the *Kalman filter* and the *particle filters*. The integration of various object features whose state is estimated by Kalman or particle filters will be examined in detail in Chapter 5.

2.1 Introduction

The main question to be addressed in this thesis refers to the inference of the states of several target features as a function of time. Both the object and background may vary throughout the sequence, follow nonlinear dynamics, suffer from deformations, and the scene might be affected by illumination changes. In order to infer all these kind of changes, most visual tracking techniques involve three basic elements, namely target representation, hypotheses generation, and the correction of these hypotheses considering some external observations.

Target representation refers to the selection of the appropriate features allowing to discriminate the target from the rest of the image. Usual modalities include geometric features, shape, motion, and appearance, such as grey scale templates or color distributions. Several visual modalities will be investigated in the next chapter.

Once a specific target feature has been represented by a parametric model, the goal of the

tracking module is to estimate the feature state throughout time by a process of generation and correction of candidate hypotheses. New hypotheses about the the target state are produced based on previous states and considering a dynamic model with a stochastic component. Subsequently, the hypotheses are corrected by the observation of some image features (for instance, if the target is represented by its contour, the observation might be the edges in the image). This process can be expressed as a temporal propagation of conditional probabilities, in terms of a Bayes filter: the *a posteriori* distribution $p(\mathbf{x}^t|\mathbf{Z}^t)$ over the target state $\mathbf{x}^t$, given the observations $\mathbf{Z}^t = \{\mathbf{z}^{t_0}, \ldots, \mathbf{z}^t\}$ up to time t, can be recursively updated according to the Bayes rule (11):

$$p(\mathbf{x}^t|\mathbf{Z}^t) = \frac{p(\mathbf{x}^t, \mathbf{Z}^{t-1}, \mathbf{z}^t)}{p(\mathbf{Z}^t)} = \frac{p(\mathbf{z}^t|\mathbf{x}^t, \mathbf{Z}^{t-1})p(\mathbf{x}^t, \mathbf{Z}^{t-1})}{p(\mathbf{Z}^t)}$$

Assuming that the observations $\mathbf{z}^{t_0}, \ldots, \mathbf{z}^t$ are conditionally independent it is satisfied that $p(\mathbf{z}^t|\mathbf{x}^t, \mathbf{Z}^{t-1}) = p(\mathbf{z}^t|\mathbf{x}^t)$, and the previous equation may be rewritten as:

$$p(\mathbf{x}^t|\mathbf{Z}^t) = \frac{p(\mathbf{z}^t|\mathbf{x}^t)p(\mathbf{x}^t, \mathbf{Z}^{t-1})}{p(\mathbf{Z}^t)}$$

Note that this is a *Markovian* dynamic model, since the state at time t only depends on the state at time $t-1$.

In order to introduce a dynamic model $p(\mathbf{x}^t|\mathbf{x}^{t-1})$, we perform the following expansion:

$$p(\mathbf{x}^t, \mathbf{Z}^{t-1}) = \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1}, \mathbf{Z}^{t-1})p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})d\mathbf{x}^{t-1} \tag{2.1}$$

Since we are also assuming that the dynamical process is independent from previous observations, Eq. 2.1 can be simplified:

$$p(\mathbf{x}^t, \mathbf{Z}^{t-1}) = \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1})p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})d\mathbf{x}^{t-1}$$

Finally, the equation that must be calculated or approximated by a tracking filter is:

$$p(\mathbf{x}^t|\mathbf{Z}^t) = \frac{p(\mathbf{z}^t|\mathbf{x}^t)\int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1})p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})d\mathbf{x}^{t-1}}{p(\mathbf{Z}^t)} \tag{2.2}$$

where the term $p(\mathbf{z}^t|\mathbf{x}^t)$ expresses the *observation density*, i.e, the probability of making observation $\mathbf{z}^t$ given that the target state at time t is $\mathbf{x}^t$. The term $p(\mathbf{x}^t|\mathbf{x}^{t-1})$ represents the *dynamic model*, i.e, the prediction of state $\mathbf{x}^t$ at time t given the previous state $\mathbf{x}^{t-1}$.

From Eq. 2.2, it can be noted that the updating procedure of Bayesian filters is performed through the two steps of hypotheses generation and hypotheses correction, previously mentioned:

- **Hypotheses generation**: Given the dynamic model $p(\mathbf{x}^t|\mathbf{x}^{t-1})$ and the a posterior distribution at the previous time step $p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})$, the state of the target is predicted according to the following update rule:

$$p(\mathbf{x}^t|\mathbf{Z}^{t-1}) = \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1})p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})d\mathbf{x}^{t-1} \tag{2.3}$$

- **Hypotheses correction**: The predicted state of the target $p(\mathbf{x}^t|\mathbf{Z}^{t-1})$ is corrected by the observation model $p(\mathbf{z}^t|\mathbf{x}^t)$:

$$p(\mathbf{x}^t|\mathbf{Z}^t) = \alpha^t p(\mathbf{z}^t|\mathbf{x}^t)p(\mathbf{x}^t|\mathbf{Z}^{t-1}) \tag{2.4}$$

  where $\alpha^t = 1/p(\mathbf{Z}^t)$ is a normalizing constant ensuring that the posterior probability over the entire state space sums up to one.

In the following sections we will briefly describe the well-known algorithms of Kalman and particle filters, as representative examples for Bayesian tracking in continuous and discrete spaces, respectively. In particular, in Chapter 5 we will use these filters to estimate and integrate the state of several object features for robust tracking purposes.

2.2 Kalman filter

In the particular case that the observation density is assumed to be Gaussian, and the dynamics are assumed to be linear with additive Gaussian noise, equations 2.3 and 2.4 result in the Kalman filter (8; 10; 30; 58; 81; 136). In the Appendix A we derive the Kalman filter equations, from the Bayesian point of view. Next, we simply include the main steps of the algorithm.

The evolution of the system (parameterized by a state vector $\mathbf{x}$) at time t is described by the dynamic model:

$$\mathbf{x}^t = \mathbf{D}^t\mathbf{x}^{t-1} + \mathbf{q}_d^t \tag{2.5}$$

where $\mathbf{D}^t$ is a square matrix denoting the deterministic component of the dynamic model, and $\mathbf{q}_d^t$ is a random variable representing the process noise, assumed to be white with normal distribution, i.e, $\mathbf{q}_d^t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_d^t)$.

The state vector is related to the observation $\mathbf{z}_t$ by the measurement equation:

$$\mathbf{z}^t = \mathbf{M}^t\mathbf{x}^t + \mathbf{q}_m^t \tag{2.6}$$

where $\mathbf{M}^t$ is a matrix denoting the deterministic component of the measurement model, and $\mathbf{q}_m^t$ is a zero-mean, white and Gaussian variable representing the measurement noise, $\mathbf{q}_m^t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_m^t)$.

Based on these models, Kalman filter updates the state vector and covariance estimates using the following 'prediction-correction' cycle:

Hypothesis generation (prediction)

$$\mathbf{x}_-^t = \mathbf{D}^t \mathbf{x}_+^{t-1} \quad \text{Predicted state vector} \tag{2.7}$$

$$\mathbf{\Sigma}_-^t = \mathbf{\Sigma}_d^t + \mathbf{D}^t \mathbf{\Sigma}_+^{t-1} (\mathbf{D}^t)^T \quad \text{Predicted state covariance} \tag{2.8}$$

Hypothesis correction

$$\mathbf{x}_+^t = \mathbf{x}_-^t + \mathbf{K}^t [\mathbf{z}^t - \mathbf{M}^t \mathbf{x}_-^t] \quad \text{State estimate} \tag{2.9}$$

$$\mathbf{\Sigma}_+^t = [\mathbb{I} - \mathbf{K}^t \mathbf{M}^t] \mathbf{\Sigma}_-^t \quad \text{State covariance estimate} \tag{2.10}$$

$$\mathbf{K}^t = \mathbf{\Sigma}_-^t (\mathbf{M}^t)^T [\mathbf{M}^t \mathbf{\Sigma}_-^t (\mathbf{M}^t)^T + \mathbf{\Sigma}_m^t]^{-1} \quad \text{Kalman Gain} \tag{2.11}$$

The subscripts '$-$' and '$+$' represent the a priori and a posteriori estimates respectively (for further details, see Appendix A).

Figure 2.1 shows in a one dimensional example, how the Kalman filter can be seen as a propagation of conditional densities process. Initially, a Gaussian distribution is available from previous iteration, and represents the probability distribution of the state vector (Fig. 2.1a). Next, a linear dynamic model with zero-mean, and white Gaussian noise is applied to this distribution, resulting in another Gaussian function (Fig. 2.1b). Finally, based on some observation, the propagated distribution is corrected, providing the a posteriori probability distribution of the state (Fig. 2.1c) .

Even though Kalman filters make strong assumptions about the nature of the dynamic model and observations, their computational efficiency has made them very popular, and useful for situations where the dynamics of the target follow paths with relatively low uncertainty. Nevertheless, in general and unconstrained situations, the observation densities involved in the tracking problem can not be adjusted to Gaussian distributions, and consequently the linear Kalman filter is not useful.

In order to handle nonlinearities both in the dynamic and observation models, several approaches based on the Kalman filter have been proposed. The *Extended Kalman Filter*

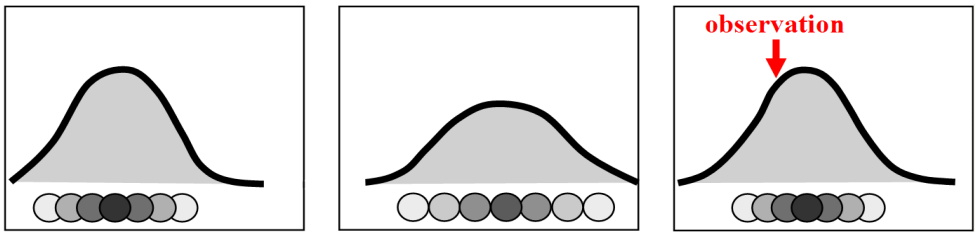
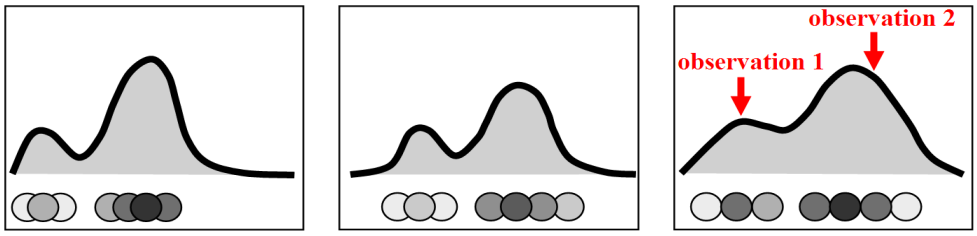


Figure 2.1: **Probability density function propagation in a Kalman filter**. (a) A posteriori probability distribution of the target state at iteration $t-1$. Although in a Kalman filter this density is assumed to be a Gaussian distribution that might be analytically represented, we depict how it would be a discretization of the Normal density, by a set of weighted particles. We represent these set of particles below the PDF plot. Their corresponding weights are proportional to the gray level, in such a way that darker tonalities represent higher weights. This kind of representation will be used in following chapters, when describing the fusion scheme. (b) In the prediction stage, the PDF of the state is propagated by a linear dynamic model with Gaussian noise. (c) Finally, in the last stage, the PDF is corrected by considering a measurement. Throughout the whole process, all the involved probability distributions are considered to be Gaussian.

(EKF) (8; 40) approximates the non linear dynamics by a Taylor expansion, and subsequently proceeds as for the linear Kalman filter. Another Kalman based approach, the *Unscented Kalman Filter* (UKF) (57; 134) proceeds by considering a set of points that are deterministically selected from $p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})$, which is assumed to have a Gaussian distribution. These points are subsequently propagated by a nonlinear dynamic model and used to re-estimate the parameters of $p(\mathbf{x}^t|\mathbf{Z}^t)$. However, the limitation of the EKF and UKF is that both of them always assume a posteriori PDF with a Gaussian distribution, which is unrealistic especially in cluttered environments.

Data association methods deal with nonlinearities in the observation model by using multiple hypotheses about the measurements. For instance, the *Probabilistic Data Association Filter* (PDAF) (8) assigns only one of a finite set of measurements to the target, based on some heuristic mechanisms. The *Joint PDAF* (JPDAF) extends the problem to the multitarget case, where each target is tracked using the same modality. The *Joint Likelihood Filter* (107), goes a step further, and allows to track different targets based on different modalities.

Nevertheless, the best results up to date, when both the dynamic and observation models do not hold the Gaussian assumption, are obtained with trackers based on particle filter formulations (3; 22; 32; 71; 83), where the Condensation algorithm (50; 51; 75), is perhaps the most



(a) (b) (c)

Figure 2.2: **Probability density function propagation in a particle filter.** (a) A posteriori probability distribution of the target state at iteration $t-1$. Note that when dealing with particle filters, the assumption of Gaussian probability distributions is no longer necessary. (b) The PDF of the state is propagated by a stochastic dynamic model. (c) Likewise in the Kalman filter procedure, the propagated PDF is corrected considering external observations. On the other side, in particle filters, multiple observations may be considered.

popular example in the computer vision community. In the following section, we will describe how the particle filters realize the Bayes filter updates.

2.3 Particle filters

Particle filters (3; 22; 32; 50; 51; 71; 75; 83) are suitable for representing the propagation of conditional densities, when the dynamic and observation probability distributions involved in the process are non-Gaussian. The main idea in particle filters is to approximate the a posteriori probability of the state by a set of random weighted samples. Subsequently, in order to simulate the Bayesian filter propagation (Eq. 2.2), the dynamic and observation models are applied individually to each one of the samples. Generally, the higher number of samples, the better approximation of the real PDF's is obtained. Fig. 2.2 depicts the stages of the propagation, which likewise in the Kalman filter, are composed of a probabilistic propagation (prediction) and a correction using some external measures (not only one, as in the Kalman filter). Next, we will explain the details of the algorithm.

The a posteriori probability $p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})$ (Fig. 2.3a) of the previous time step is approximated by a set of n weighted samples $\{\mathbf{s}_i^{t-1}, \pi_i^{t-1}\}$, $i = 1, \ldots, n$ where $\pi_i^{t-1} \in [0, 1]$ is the weight for particle $\mathbf{s}_i^{t-1}$. The whole set of weights is normalized to sum up to one, i.e, $\sum_{i=1}^{n} \pi_i = 1$. Given this representation, the steps to estimate the a posteriori probability $p(\mathbf{x}^t|\mathbf{Z}^t)$ can be summarized as follows:

Resampling: A common problem in particle filters refers to a degeneracy phenomenon, where all but one particle will have negligible weight after a few iterations (3). This effect may be reduced using a resampling stage at the beginning of each iteration, consisting in the elimination of those particles having smaller weights and concentrating on particles with larger weights. That is, the set $\{\mathbf{s}_i^{t-1}, \pi_i^{t-1}\}$, $i = 1, \ldots, n$, is resampled (sampling with replacement) according to the weights π_i^{t-1}, in order to define a new set $\{\tilde{\mathbf{s}}_i^{t-1}, \tilde{\pi}_i^{t-1}\}$.

The sampling with replacement is simulated on the basis of the following deterministic algorithm (a different algorithm based on a random sampling is explained in (50)):

Deterministic resampling algorithm: Given the set $\{\mathbf{s}_i^t, \pi_i^t\}$, $i = 1, \ldots, n$, the cumulative probabilities are calculated from π_i^t:

$$c_i^{t_0} = 0$$
$$c_i^t = \sum_{j=1}^{i} \pi_i^t$$

Subsequently, the new particle set $\{\tilde{\mathbf{s}}_i^t, \tilde{\pi}_i^t\}$, $i = 1, \ldots, n$ is defined as follows:

$$\tilde{\mathbf{s}}_i^t = \mathbf{s}_{\epsilon(i)}^t \qquad \text{where } \epsilon(i) \text{ is the smallest } j \text{ such that } c_j^t \geq \frac{i}{n}$$
$$\tilde{\pi}_i^t = \frac{1}{n}$$

The result of this resampling phase can be observed in Fig. 2.3b. Note that the particles with higher weight can be selected several times, and therefore, identical copies of these elements may appear in the resampled set. On the other hand, some particles having low weight may not be chosen at all, and are extinguished of the process. Observe that all particle weights are identical at this point of the iteration.

Hypotheses generation (prediction): Each element of the set $\{\tilde{\mathbf{s}}_i^{t-1}, \tilde{\pi}_i^{t-1}\}$, $i = 1, \ldots, n$ is propagated according to a stochastic dynamic model, represented as a conditional density $p(\mathbf{x}^t|\mathbf{x}^{t-1})$. A new set $\{\mathbf{s}_i^t, \tilde{\pi}_i^{t-1}\}$, $i = 1, \ldots, n$ is generated, where $\mathbf{s}_i^t \sim p(\mathbf{s}_i^t|\mathbf{s}_i^{t-1})$. In this stage the weights associated to each particle are not modified (Fig. 2.3c).

Hypotheses correction: The stochastic resampling and propagation operations performed in the two previous stages provide an approximation of the likelihood term $p(\mathbf{x}^t|\mathbf{Z}^{t-1}) = \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1}) p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1}) d\mathbf{x}^{t-1}$ of the Bayes filter equation (Eq. 2.2). The subsequent action to realize consist of modifying the particle weights based on the observation density $p(\mathbf{z}^t|\mathbf{x}^t)$, allowing to obtain the set $\{\mathbf{s}_i^t, \pi_i^t\}$, $i = 1, \ldots, n$, which approximates the a posteriori state density at time t, that is, $p(\mathbf{x}^t|\mathbf{Z}^t)$. Weights π_i^t are computed as:

$$\pi_i^t \propto p(\mathbf{z}^t|\mathbf{s}_i^t) \tilde{\pi}_i^{t-1}$$

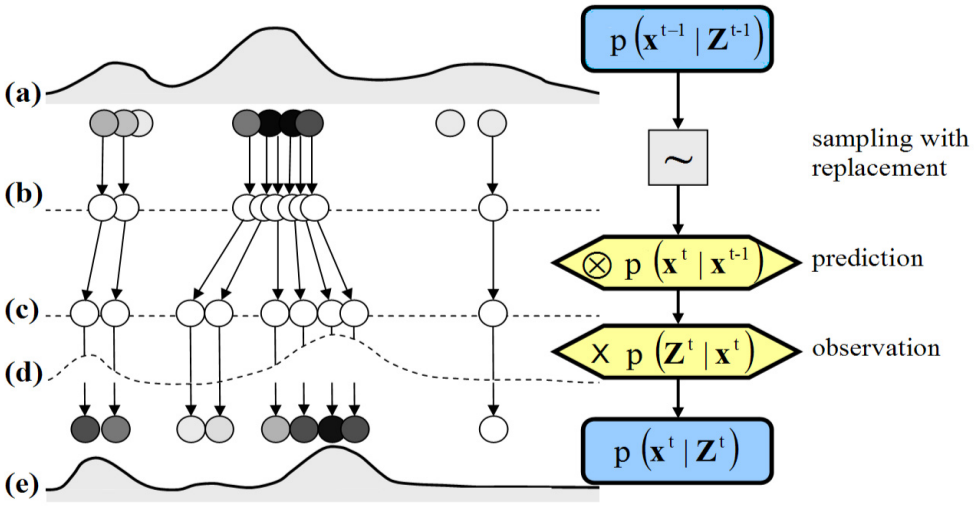


Figure 2.3: **One iteration of a particle filter algorithm.** (a) $p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})$: A posteriori probability of the state at iteration $t-1$, which is the input of the filter at iteration t. As it is commented in the text, the PDF is approximated by a set of weighted particles. (b) Resampling operation. Note that the number of samples is kept constant, and some particles having relatively low weights are extinguished, while other particles with high weight are chosen several times. (c) Sample propagation through a stochastic dynamic model $p(\mathbf{x}^t|\mathbf{x}^{t-1})$. The set of propagated samples, approximate the distribution $p(\mathbf{x}^t, \mathbf{Z}^{t-1})$. At this point, the particles are not weighted yet. (d) Observation function $p(\mathbf{z}^t|\mathbf{x}^t)$. (e) The propagated samples are weighted according to the observation function. This set of weighted particles approximates the a posteriori distribution of the state vector, at iteration t, that is, $p(\mathbf{x}^t|\mathbf{Z}^t)$. On the right side of the figure, the main steps of the particle filter algorithm are depicted by using the diagram adapted from (75), where the symbol $\sim$ denotes the resampling operation, $*$ represents a convolution operation by the dynamic model, and $\times$ is the multiplication by the observation density.

where $p(\mathbf{z}^t|\mathbf{s}_i^t)$ is a non-negative function representing the observation function (Fig. 2.3d).

Estimate: Although the prediction and correction stages are the main phases in a particle filter iteration, it is common to include an extra final step, where the set of n samples $\{\mathbf{s}_i^t, \pi_i^t\}$ is used to compute some estimate about the target state. For instance, a mean state of the target might be calculated according to:

$$E(\mathbf{x}^t) = \sum_{i=1}^{n} \pi_i^t \mathbf{s}_i^t$$

2.4 Summary

In this chapter we have established the bases to analyze the tracking problem as a propagation of conditional densities, which can be formulated by means of a Bayesian filter. From this point of view, we have introduced the basic formulation of two recursive Bayesian filters which will be used in following chapters: the Kalman filter, and particle filters. The former assumes that all the probabilities involved in the process follow Gaussian distributions and linear dynamics with Gaussian noise, while the latter allows to generalize the problem for non-Gaussian distributions and non-linear dynamics. However, the procedure of both filters can be identically separated in two stages. First, a prediction of the target state is formulated based on the target state at the previous time step and considering a stochastic dynamic model. Subsequently, according to some external observation, the predicted distribution is corrected.

All these concepts have been presented from a theoretical perspective. In the following chapters, we will define each one of the filter components, in order to design a robust tracking system. In Chapter 3, the state vectors associated to several target cues will be described. In Chapter 5, we will concentrate on the dynamic and observation models of the individual features. Furthermore, in Chapter 5, we will propose an integration scheme, for integrating several filters (either Kalman and particle filters) into a same framework, which will allow to develop a robust tracking algorithm.

Chapter 3

Visual features for robust tracking

This chapter describes the visual features that will be used to characterize the target robustly. Specifically, the object will be represented by appearance and geometric cues. The main contribution of this chapter refers to the use of an object dependent colorspace (we call it *Fisher* colorspace), which maximizes the distance between the target and background color representation. This is a desirable property for any tracking system. Therefore, we will consider the actual colorspace, and the color representation in such colorspace, as appearance features. Geometric cues will be described by a snake approximating the object contour and a Kalman predicted bounding box surrounding the object, which provides a rough estimation about its position.

It is important to note that the present chapter, will only describe each one of the features. In Chapter 5 all these cues will be integrated in a probabilistic framework for tracking purposes, where the state of the features will be estimated by the Kalman and the particle filters described in the previous chapter.

3.1 Introduction

Similar to any pattern classification task, visual tracking needs to address the initial issue about the selection of the appropriate features to represent the target and allow to discriminate it from the rest of the image. Since tracking applications search for efficient computational algorithms, the selected features might be represented by concise parameterizations. Furthermore, these cues should be robust to several artifacts present in real and unconstrained environments, such as cluttered backgrounds, non-stationary lighting conditions or non-linear target dynamics. By robustness, it is meant that the feature state remains constant or is adapted to these

non-stationary situations of the scene.

In the literature, the visual modules most commonly used in tracking tasks include geometric cues, motion, color, contrast, texture, appearance and shape. For instance, the separation of various regions in video sequences, according to different motion models is considered by Torr (127), and Torr and Zisserman (128). These works are based on an initially selected set of image features (usually points of interest) whose correspondences in consecutive images can be robustly determined. Based on the correspondence of this reduced set of points, an error measure is minimized and different movement models are identified. Each one of the pre-selected pixels is classified to a specific motion class permiting to group and recognize image regions having similar velocities. Other approaches minimize an error measure based on information collected from all pixels in the images (not only a reduced set of points). For instance, Irani and Anandan (49) employ a brightness constraint of all image points to identify the motion groups present in a video sequence. It is assumed that the dominant motion corresponds to the background points, and the target is identified by detecting image pixels that do not fit the dominant motion. A similar work was previously presented by Black and Anandan (16).

Color distributions are also often used as target representations. A simple non-parametric method for modeling color distributions are color histograms. For instance, Birchfield (13) approximates the target color by a histogram, and then, the object of interest is localized in successive frames by searching image regions with a similar histogram to that of the target. The measure of similarity is based on a histogram intersection metric proposed by Swain and Ballard (123). Sigal *et al.* (119) use a priori learned skin-like color histograms to classify all image pixels into skin and no-skin classes. In other approaches, for instance Raja *et al.* (109), and Yang *et al.* (144), the color distribution is represented by a parametric model using *Mixture of Gaussians*. A completely different representation for color distributions is presented by Wu and Huang (141), modeling target color by a SASOM (*Structure Adaptative SOM*) neural network, whose structure is learned by an offline training.

Contour is another feature commonly used to characterize the target, basically because it offers more robustness to smooth changes of illumination than appearance-based cues. The tracking algorithms proposed by Isard and Blake (51), or MacCormick (75), are examples of object trackers based on contours parameterized by B-splines.

Target shape can also be modeled by 3D meshes. For instance, Kuch and Huang (66) represent the surface of the target by a spline-based surface. Nevertheless, these methods require a high number of parameters and control points to be specified. Other approaches approximate

the parts of an articulated target by geometric shapes parameterized by a reduced number of parameters, such as in (120), where a human body is approximated and tracked by generalized cylinders.

This brief description of existing techniques refers uniquely to the tracking methods that represent the target using a single feature. To provide a more robust description of the object, several features might be simultaneously considered. Multiple cue integration will be covered in the following chapters (Chapter 4 and 5). In the present chapter we center our attention to the description of the individual cues.

Specifically, in this thesis we will describe the object using both appearance and geometric cues. The first appearance cue that will be considered, is an object dependent colorspace, named *Fisher* colorspace. In almost all of the previous approaches, no attention has been paid to the selection of the colorspace where the object is represented. Instead, in this dissertation we propose to represent the color distribution of the target in the Fisher colorspace, which maximizes the distance of the target colorpoints with respect to background colorpoints. This colorspace is dependent on the object appearance, and consequently, can be considered as a target feature. The distribution of the colorpoints onto the Fisher plane will be parameterized by a Mixture of Gaussians model, fitted to the data using *Expectation-Maximization* (EM).

Furthermore, the object will be represented by its contour, approximated by a snake model. This model, allows to deal with local and non-rigid deformations of the object.

Besides all these features, we will include a position prediction module which, based on the Kalman filter, will provide a rough estimate of the position of a bounding box surrounding the object. In the following sections we will describe in detail each one of the features.

3.2 Colorspace selection

An important initial issue for any color-based tracking or figure-ground segmentation algorithm, concerns the selection of the colorspace where the data will be represented. Nevertheless, throughout the literature there is not a clear consensus about which colorspace to use. A good review about different colorspaces, may be found in (38) and (121). Next, we just overview the principal models.

3.2.1 Existing colorspaces

The most popular method to encode the color information is the *RGB* model, where the color is represented by three components, resulting from the response of three separate photoreceptors:

$$R = \int_\lambda \sigma_R(\lambda)E(\lambda)d\lambda \qquad G = \int_\lambda \sigma_G(\lambda)E(\lambda)d\lambda \qquad B = \int_\lambda \sigma_B(\lambda)E(\lambda)d\lambda \tag{3.1}$$

where σ_k, with $k = \{R, G, B\}$ is the sensitivity of the k-th type of receptor, $E(\lambda)$ is the light arriving to the receptor and λ are the wavelengths. The set of available colors is represented in the cube of Fig. 3.1a. The axis connecting the corners corresponding to the white and black color (gray level axis), defines the intensity of the color. As we will mention in the following section, a desirable property of a colorspace is to exhibit certain invariance in the presence of illumination changes. This may be accomplished by the normalized version of the 'full' colorspaces. For instance, in the case of the RGB colorspace, its normalized version would be the rgb colorspace:

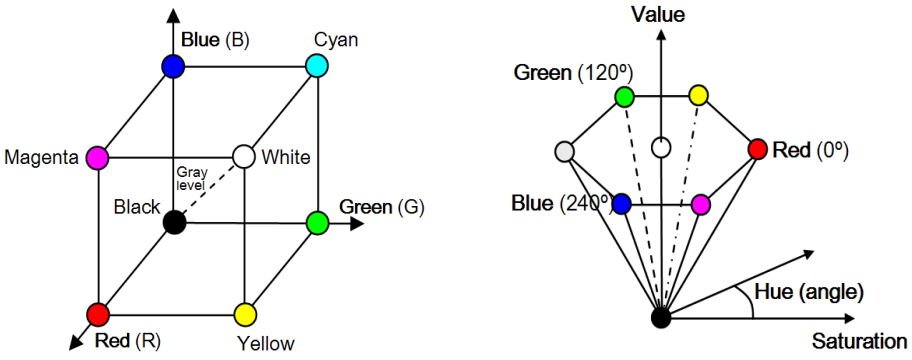
$$r = \frac{R}{R+G+B} \qquad g = \frac{G}{R+G+B} \qquad b = \frac{B}{R+G+B} \tag{3.2}$$

Since $r + g + b = 1$, the normalized rgb can be represented by only considering two of the components.

Other color models are inspired by human color perception. According to Foley *et al.* (37), the human vision system uses three measurements to represent color, namely *hue*, *saturation*, and *brightness* . *Hue* describes the 'pure' most similar color the light is perceived to be. *Saturation* describes the distance of the color with respect to a gray with identical intensity. *Brightness* refers to the perceived achromatic luminance of the light. Based on this description, it is defined the *HSV* (Hue, Saturation and Value) color model, where each one of the components is expressed as a function of the R, G, B components:

$$\begin{aligned} H(R,G,B) &= \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right) \\ S(R,G,B) &= 1 - \frac{\min(R,G,B)}{R+G+B} \\ V(R,G,B) &= R+G+B \end{aligned} \tag{3.3}$$

If the *RGB* cube of Fig. 3.1a is seen from the gray level axis point of view, it allows to define the parameters of the *HSV* colorspace. This is shown in Fig. 3.1b.



(a) (b)

Figure 3.1: **Representation of the *RGB* and *HSV* colorspaces** (a) *RGB* color cube. (b) *HSV* color representation. Note that the parameters of the *HSV* colorspace can be determined seeing the *RGB* color cube from the point of view of the gray level line.

Since the RGB colorspace excludes some colors of the visible spectrum and is too dependent on the sensor features, the *CIE* (*Commission Internationale d'Eclairage*) defined the *CIE XYZ* color model, which can be produced from *RGB* coordinates by an empirically computed linear transformation. For instance, the matrix for a NTSC receiver system is:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.607 & 0.174 & 0.200 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.066 & 1.116 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.4}$$

and the corresponding normalized components:

$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = \frac{Z}{X+Y+Z} \tag{3.5}$$

From the coordinates *XYZ*, other *CIE* color models can be constructed. For instance, the *CIE Lab*, which is expressed as:

$$\begin{aligned} L &= 25\,(100Y/Y_0)^{1/3} - 16 \\ a &= 500\left[(X/X_0)^{1/3} - (Y/Y_0)^{1/3}\right] \\ b &= 200\left[(Y/Y_0)^{1/3} - (Z/Z_0)^{1/3}\right] \end{aligned} \tag{3.6}$$

where (X_0, Y_0, Z_0) are the X, Y and Z coordinates of a reference white patch. Many more colorspaces can be defined basically as linear transformations of the RGB or XYZ coordinates, such as the *YUV* and *YIQ* models, which are utilized in the European and American TV

color system, respectively:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.437 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.7}$$

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.8}$$

In (126) nine different colorspaces are compared for skin detection tasks.

3.2.2 Desirable colorspace properties

Among the various colorspaces just described, none has a clear advantage over the others. This is the reason why, within the visual tracking literature, almost all previously described colorspaces have been indistinctly utilized. For instance, the *RGB* colorspace is used in (100; 108; 124). The efforts described in (25; 114; 140; 145) represent color by the normalized RGB model. Maybe the most extensively used colorspace is the *HSV* (109; 119; 122; 131; 141), and a two dimensional version considering only the *HS* chromaticities components (48; 64; 102). Some approaches (61; 139) are based on the *YUV* colorspace as well.

All this variety in the types of color models is due to the fact that there does not exist a criterion for the selection of the appropriate colorspace. In most of the previously cited approaches, the selection of the colorspace to use is based on some prior empirical experiments, where the color model is selected by a trial and error procedure among the various available colorspaces.

In contrast to previous approaches, in this thesis we propose to select the colorspace using specific criteria focused on visual tracking applications. The premises that will define the utilized colorspace are the following:

1. Since the first goal of visual tracking is to discriminate the object of interest from the rest of the scene, an important function of the colorspace should be to maximize the distance between target and background color distributions.

2. Furthermore, in order to deal with dynamic environments suffering from non-stationary lighting conditions, the colorspace should demonstrate a certain degree of invariance to illumination changes, that is, the optimal colorspace maximizing the foreground/background separation should not be affected by appearance changes produced by illumination changes.

Unfortunately, none of the existing colorspaces is tailored to satisfy the above criteria. At most, those colorspaces involving some normalization of the *R*, *G*, *B* components, such as normalized *RGB* or normalized *XYZ*, show invariance in the colorpoints representation when images are affected by illumination changes (uniform scaling and shifting of the light). Nevertheless, these colorspaces do not guarantee the separation of the foreground and background classes, which is extremely important for tracking tasks.

In the following section, we will propose the use of a colorspace dependent on the target and background appearances, which satisfies both previous criteria.

3.2.3 Fisher colorspace

In order to accomplish the first of the criteria previously mentioned, we will make use of pattern recognition techniques. The representation of the target and background colorpoints through a color model maximizing the separation of both classes may be analyzed as a standard classification problem based on *Discriminant Analysis*. In particular we are interested in the linear techniques (Linear Discriminant Analysis -LDA-, also called Fisher Discriminant Analysis (33; 39)). From this point of view our problem may be reduced to the search of the hyperplane (Fisher plane) that best separates the two classes. The following section will show that this linear transformation offers certain robustness to illumination changes (which is the second criterion that we have previously defined for colorspace selection).

Next, the procedure for determining the Fisher colorspace will be explained in detail, which will actually be the original *RGB* data projected onto the Fisher plane. This colorspace may be computed from a single training image, acquired with a *RGB* camera and where the points belonging to the object and the points belonging to the background are provided by the user.

Assume that the set of n image pixels are arranged into a $n \times 3$ matrix $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_n]^T$, where the rows represent the individual pixels and the columns correspond to each of the color channels in the RGB colorspace. $n_{\mathcal{O}}$ of these pixels belong to the object $\mathcal{O}$, represented by the matrix $\mathbf{C}_{\mathcal{O}} = [\mathbf{c}_{\mathcal{O},1}, \ldots, \mathbf{c}_{\mathcal{O},n_{\mathcal{O}}}]^T$ and the rest of $n_{\mathcal{B}}$ pixels $\mathbf{C}_{\mathcal{B}} = [\mathbf{c}_{\mathcal{B},1}, \ldots, \mathbf{c}_{\mathcal{B},n_{\mathcal{B}}}]^T$ belong to the background $\mathcal{B}$. We wish to determine which plane is the most effective in discriminating between these two subsets of points. Let us denote such a plane by $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2]^T \in \mathbb{R}_{2\times 3}$, where $\mathbf{w}_1$ and $\mathbf{w}_2$ are vectors in the *RGB* space, spanning the points lying on the plane. The projection of $\mathbf{C}_{\mathcal{O}}$ and $\mathbf{C}_{\mathcal{B}}$ onto this plane, generates the sets $\mathbf{F}_{\mathcal{O}} = \mathbf{C}_{\mathcal{O}}\mathbf{W}^T \in \mathbb{R}_{n_{\mathcal{O}}\times 2}$ and $\mathbf{F}_{\mathcal{B}} = \mathbf{C}_{\mathcal{B}}\mathbf{W}^T \in \mathbb{R}_{n_{\mathcal{B}}\times 2}$, respectively.

The goal of Fisher Discriminant Analysis is to find the best orientation of the plane $\mathbf{W}$, such that the separation of the projected subsets $\mathbf{F}_{\mathcal{O}}$ and $\mathbf{F}_{\mathcal{B}}$ is maximized. In order to determine such a plane, Fisher-LDA considers the maximization of the following objective function:

$$J(\mathbf{W}) = \frac{\mathbf{W}\mathbf{S}_b\mathbf{W}^T}{\mathbf{W}\mathbf{S}_w\mathbf{W}^T} \tag{3.9}$$

where $\mathbf{S}_w$ is the *within class scatter matrix* and $\mathbf{S}_b$ is the *between class scatter matrix*. These matrices are defined as:

$$\mathbf{S}_w = \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \sum_{i=1}^{n_\varepsilon} (\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)(\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)^T \tag{3.10}$$

$$\mathbf{S}_b = \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} n_\varepsilon (\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})(\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})^T \tag{3.11}$$

where,

$$\bar{\mathbf{c}}_\varepsilon = \frac{1}{n_\varepsilon} \sum_{i=1}^{n_\varepsilon} \mathbf{c}_{\varepsilon,i} \qquad \text{is the } \varepsilon\text{-class mean vector} \tag{3.12}$$

$$\bar{\mathbf{c}} = \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \bar{\mathbf{c}}_\varepsilon \qquad \text{is the total mean vector} \tag{3.13}$$

Note that the maximization of the criterion in Eq. 3.9 makes sense in that it searches for the separation of the class means in the projected space (high $\mathbf{S}_b$), while at the same time the classes remain compact (small $\mathbf{S}_w$). The classic Fisher-LDA method (33) maximizes the J objective function by constructing the rows of $\mathbf{W}$ with the eigenvectors of $\mathbf{S}_w^{-1}\mathbf{S}_b$ having the highest eigenvalues.

Nevertheless, this approach has a limitation. In the general case of a $\mathcal{C}$-class problem, $\mathbf{S}_b$ will be the sum of $\mathcal{C}$ matrices of rank one, and since only $\mathcal{C}-1$ of these matrices are independent (see Eqs. 3.11 and 3.13), the rank of $\mathbf{S}_b$ will be at most $\mathcal{C}-1$. As a consequence the rank of $\mathbf{S}_w^{-1}\mathbf{S}_b$ will be $\mathcal{C}-1$ as well, and $\mathbf{S}_w^{-1}\mathbf{S}_b$ will have only $\mathcal{C}-1$ nonzero eigenvalues. Subsequently, the hyperplane $\mathbf{W}$ will be spanned at most by $\mathcal{C}-1$ eigenvectors.

In the two class problem discussed here, this would mean that the *RGB* points would be projected on a hyperplane of dimension one, i.e, a line. As will be discussed in the following section, the projection of the data onto a linear space can increase robustness to illumination changes. Therefore, the projection of the original data onto a line would satisfy these requirements, but with the cost of losing too much information. A better choice, consists of projecting the *RGB* data onto a plane.

In order to get a 2D discrimination plane, even in the case of 2 classes, we need to use the nonparametric version of Linear Discriminant Analysis (39). The key point of this LDA extension is that it computes the between class scatter matrix $\mathbf{S}_b$ using local information and the *K-Nearest Neighbors* (KNN) rule, which allows to obtain a full rank matrix. For the object ($\mathcal{O}$) and background ($\mathcal{B}$) classes of our particular problem, the *nonparametric between class scatter matrix* (denoted $\mathbf{\Sigma}_b$) is defined as,

$$\begin{aligned}\mathbf{\Sigma}_b = &\frac{1}{n}\sum_{i=1}^{n_{\mathcal{O}}} w_i \left(\mathbf{c}_{\mathcal{O},i} - M_{\mathcal{B}}^k(\mathbf{c}_{\mathcal{O},i})\right)\left(\mathbf{c}_{\mathcal{O},i} - M_{\mathcal{B}}^k(\mathbf{c}_{\mathcal{O},i})\right)^T \\ &+ \frac{1}{n}\sum_{i=1}^{n_{\mathcal{B}}} w_i \left(\mathbf{c}_{\mathcal{B},i} - M_{\mathcal{O}}^k(\mathbf{c}_{\mathcal{B},i})\right)\left(\mathbf{c}_{\mathcal{B},i} - M_{\mathcal{O}}^k(\mathbf{c}_{\mathcal{B},i})\right)^T\end{aligned} \tag{3.14}$$

where $M_{\varepsilon}^k(\mathbf{c}_i)$ is the mean of the k nearest neighbors in class $\varepsilon = \{\mathcal{O}, \mathcal{B}\}$ to a point $\mathbf{c}_i$, and w_i is a weighting function for deemphasizing samples far from the classification boundary (39).

We conclude with the main steps of the algorithm, which given two sets $\{\mathbf{c}_{\mathcal{O},1}, \cdots, \mathbf{c}_{\mathcal{O},n_{\mathcal{O}}}\}$, $\{\mathbf{c}_{\mathcal{B},1}, \cdots, \mathbf{c}_{\mathcal{B},n_{\mathcal{B}}}\}$ of RGB pixel values used as training data, obtains their optimum linear mapping onto a 2D plane:

1. Calculate the *within scatter matrix* $\mathbf{S}_w$ based on Eq. 3.10
2. Transform the data so that the averaged covariance matrix $\mathbf{S}_w$, becomes the identity matrix (39). This can be achieved by whitening the original data with respect to $\mathbf{S}_w$. That is, transform $\mathbf{c}$ to $\mathbf{d} = \mathbf{\Lambda}^{-1/2}\mathbf{\Omega}^T\mathbf{c}$, where $\mathbf{\Lambda}$ and $\mathbf{\Omega}$ are the 3×3 eigenvalue and eigenvector matrices of $\mathbf{S}_w$.
3. Select k and (in the D-space, i.e, the space of $\mathbf{d}$ data) compute $\mathbf{\Sigma}_b$ using Eq. 3.14.
4. Select the two eigenvectors ψ_1, ψ_2 of $\mathbf{\Sigma}_b$ with the two largest eigenvalues. These vectors are arranged into the matrix $\mathbf{\Psi} = [\psi_1, \psi_2] \in \mathbb{R}_{3\times 2}$.
5. The optimum linear mapping from the original RGB space to the discriminant subspace (we call it *Fisher colorspace*) is given by $\mathbf{f} = \mathbf{\Psi}^T\mathbf{\Lambda}^{-1/2}\mathbf{\Omega}^T\mathbf{c}$, where $\mathbf{c} \in \mathbb{R}_{3\times 1}$ is a point represented in the *RGB* colorspace and $\mathbf{f} \in \mathbb{R}_{2\times 1}$ is its projection onto the Fisher plane.

   Therefore, the Fisher plane can be written as:

$$\mathbf{W} = \mathbf{\Psi}^T\mathbf{\Lambda}^{-1/2}\mathbf{\Omega}^T \tag{3.15}$$

(a) (b)

Figure 3.2: ***RGB* color distribution of a scene.** (a) Image of the scene. (b) Color distribution of the image pixels, represented in the *RGB* colorspace.

Figures 3.2 and 3.3 depict the main steps to compute the Fisher plane, for two different targets in the same image. Fig. 3.2a shows the scene and in Fig. 3.2b all the image pixels are represented in the *RGB* colorspace. The main idea behind the Fisher colorspace is that for color-based tracking tasks we can choose a colorspace dependent on the target appearance, which maximizes its distance (in colorspace coordinates) with respect to the color of the rest of image points. For example, if we wish to track the ladybird (Fig. 3.3a), the colorspace can be selected offline as a simple calibration procedure before the tracking stage. The points belonging to the object and background are initially provided (Fig. 3.3b), and the Fisher plane is computed based on the non-parametric LDA explained previously (Fig. 3.3c). The separation of the target and background colorpoints projected onto this plane is maximized (Fig. 3.3d,e). In the case that the flower petals were the tracked object, the system would be trained according to the equivalent stages indicated in Figures 3.3f- 3.3j. Note in Fig. 3.4, the difference of the Fisher planes corresponding to the different targets.

3.2.4 Fisher colorspace in the presence of lighting changes

One desirable property that a colorspace should satisfy, is to provide a representation of the scene invariant to illumination changes, i.e, when the lighting conditions change, an optimal colorspace would be one where the color representation of the objects in the image were maintained constant, or the mapping between the images under different illuminants could be learned and used to correct the illumination changes. This is precisely the subject of interest of the color constancy algorithms (6; 7; 34). Unfortunately, since this is a extremely challenging task, available methodologies are generally constrained to laboratory and artificial lighting

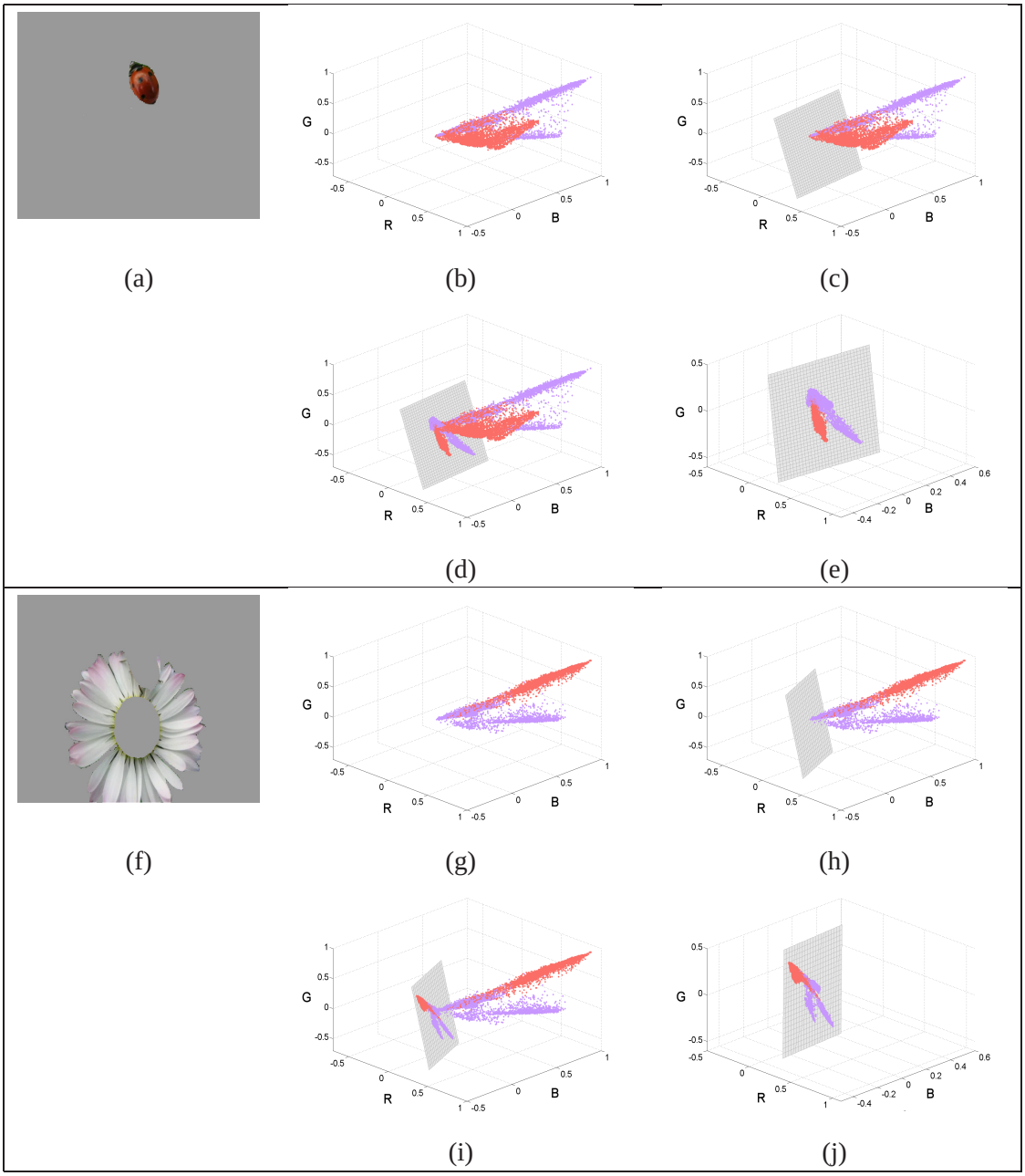


Figure 3.3: **Stages for determining the Fisher plane for two different targets.** (a)-(e): Fisher colorspace for tracking the ladybird. (a) Hand segmented target image. (b) Manual separation of the image pixels into the foreground (red) and background (blue) classes. (c) Fisher plane adjusted to the training classes. (d) Projection of the foreground and background points onto the Fisher plane. (e) Detail of the projected points. (f)-(j): Fisher colorspace for tracking the flower petals. These figures have an equivalent meaning as figures (a)-(e).

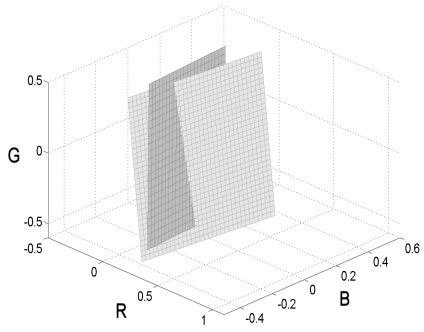


Figure 3.4: **Fisher planes for different targets.** Detail of the Fisher planes associated to the targets commented in Fig. 3.3. Note that the Fisher plane is 'object-dependent'.

conditions.

As an alternative to the search of a color representation invariant to distinct illuminants, in this thesis we propose to adapt both the colorspace and the colorpoints representation through time (this will be discussed in Chapter 5).

Nevertheless, it is important to note that the Fisher colorspace is invariant to certain lighting effects, such as uniform scaling and shifting of the image colorpoints. This invariace will relax considerably the complexity of the colorspace adaptation process. Next we will show how the Fisher colorspace remains stationary in the presence of the illumination artifacts just mentioned:

Without loss of generality, in the following proofs and derivations we will assume the parametric version of the Fisher plane, described by the maximization of the objective function $J(\mathbf{W}) = (\mathbf{W}\mathbf{S}_b\mathbf{W}^T)(\mathbf{W}\mathbf{S}_w\mathbf{W}^T)^{-1}$ where $\mathbf{S}_b$ and $\mathbf{S}_w$ are defined by Eqs. 3.10 and 3.11, respectively.

Lemma 1. *The Fisher plane is invariant to uniform illumination scaling*

Proof: Given all image points $\mathbf{c}_i, i = 1, \ldots, n$ represented in the *RGB* colorspace, a uniform illumination scaling is defined by the mapping:

$$\mathcal{S}: \quad \mathbf{c}_i \rightarrow \alpha\mathbf{c}_i \tag{3.16}$$

where α is the scaling factor.

We assume that the classification of the image points into the object ($\mathcal{O}$) and background ($\mathcal{B}$) classes is available. Under these circumstances, the following statements about the total mean and the class means are satisfied:

$$\begin{aligned}
\mathcal{S}(\bar{\mathbf{c}}_\varepsilon) &= \mathcal{S}\left(\frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\mathbf{c}_{\varepsilon,i}\right) = \frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\alpha\mathbf{c}_{\varepsilon,i} \\
&= \alpha\frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\mathbf{c}_{\varepsilon,i} = \alpha\bar{\mathbf{c}}_\varepsilon \qquad (3.17) \\
\mathcal{S}(\bar{\mathbf{c}}) &= \mathcal{S}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\bar{\mathbf{c}}_\varepsilon\right) = \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\mathcal{S}(\bar{\mathbf{c}}_\varepsilon) \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\alpha\bar{\mathbf{c}}_\varepsilon = \alpha\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\bar{\mathbf{c}}_\varepsilon = \alpha\bar{\mathbf{c}} \qquad (3.18)
\end{aligned}$$

As a consequence, the transformed within class scatter matrix and between class scatter matrix may we written as

$$\begin{aligned}
\mathcal{S}(\mathbf{S}_w) &= \mathcal{S}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\sum_{i=1}^{n_\varepsilon}(\mathbf{c}_{\varepsilon,i}-\bar{\mathbf{c}}_\varepsilon)(\mathbf{c}_{\varepsilon,i}-\bar{\mathbf{c}}_\varepsilon)^T\right) \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\sum_{i=1}^{n_\varepsilon}(\mathcal{S}(\mathbf{c}_{\varepsilon,i})-\mathcal{S}(\bar{\mathbf{c}}_\varepsilon))(\mathcal{S}(\mathbf{c}_{\varepsilon,i})-\mathcal{S}(\bar{\mathbf{c}}_\varepsilon))^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\sum_{i=1}^{n_\varepsilon}(\alpha\mathbf{c}_{\varepsilon,i}-\alpha\bar{\mathbf{c}}_\varepsilon)(\alpha\mathbf{c}_{\varepsilon,i}-\alpha\bar{\mathbf{c}}_\varepsilon)^T \\
&= \alpha^2\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\sum_{i=1}^{n_\varepsilon}(\mathbf{c}_{\varepsilon,i}-\bar{\mathbf{c}}_\varepsilon)(\mathbf{c}_{\varepsilon,i}-\bar{\mathbf{c}}_\varepsilon)^T \\
&= \alpha^2\mathbf{S}_w \qquad (3.19) \\
\mathcal{S}(\mathbf{S}_b) &= \mathcal{S}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}n_\varepsilon(\bar{\mathbf{c}}_\varepsilon-\bar{\mathbf{c}})(\bar{\mathbf{c}}_\varepsilon-\bar{\mathbf{c}})^T\right) \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}n_\varepsilon(\mathcal{S}(\bar{\mathbf{c}}_\varepsilon)-\mathcal{S}(\bar{\mathbf{c}}))(\mathcal{S}(\bar{\mathbf{c}}_\varepsilon)-\mathcal{S}(\bar{\mathbf{c}}))^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}n_\varepsilon(\alpha\bar{\mathbf{c}}_\varepsilon-\alpha\bar{\mathbf{c}})(\alpha\bar{\mathbf{c}}_\varepsilon-\alpha\bar{\mathbf{c}})^T \\
&= \alpha^2\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}n_\varepsilon(\bar{\mathbf{c}}_\varepsilon-\bar{\mathbf{c}})(\bar{\mathbf{c}}_\varepsilon-\bar{\mathbf{c}})^T \\
&= \alpha^2\mathbf{S}_b \qquad (3.20)
\end{aligned}$$

Finally, the objective function to be maximized in order to compute the Fisher plane, is:

$$\begin{aligned} \mathcal{S}(J(\mathbf{W})) &= \mathcal{S}\left(\frac{\mathbf{W}\mathbf{S}_b\mathbf{W}^T}{\mathbf{W}\mathbf{S}_w\mathbf{W}^T}\right) = \frac{\mathbf{W}\mathcal{S}(\mathbf{S}_b)\mathbf{W}^T}{\mathbf{W}\mathcal{S}(\mathbf{S}_w)\mathbf{W}^T} \\ &= \frac{\mathbf{W}\alpha^2\mathbf{S}_b\mathbf{W}^T}{\mathbf{W}\alpha^2\mathbf{S}_w\mathbf{W}^T} = \frac{\mathbf{W}\mathbf{S}_b\mathbf{W}^T}{\mathbf{W}\mathbf{S}_w\mathbf{W}^T} = J(\mathbf{W}) \end{aligned} \tag{3.21}$$

Thus, the criteria used to compute the Fisher plane for two images related by a linear scaling are exactly the same. Therefore, we conclude that the Fisher plane is invariant to uniform illumination scaling. ■

Lemma 2. *The Fisher plane is invariant to a uniform illumination shifting*

Proof: The proof for this lemma is quite similar to the proof for lemma 1. In this case, given all image points $\mathbf{c}_i, i = 1, \ldots, n$ represented in the *RGB* colorspace, a uniform lighting shifting is defined by the mapping:

$$\mathcal{T}: \quad \mathbf{c}_i \rightarrow \mathbf{c}_i + \beta \tag{3.22}$$

where β is the shifting factor.

We assume again that the classification of the image points into the object ($\mathcal{O}$) and background ($\mathcal{B}$) classes is available. Subsequently, the class means and the total mean for the lighting shifted image can be expressed as:

$$\begin{aligned} \mathcal{T}(\bar{\mathbf{c}}_\varepsilon) &= \mathcal{T}\left(\frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\mathbf{c}_{\varepsilon,i}\right) = \frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\mathbf{c}_{\varepsilon,i} + \beta \\ &= \frac{n_\varepsilon}{n_\varepsilon}\beta + \frac{1}{n_\varepsilon}\sum_{i=1}^{n_\varepsilon}\mathbf{c}_{\varepsilon,i} = \bar{\mathbf{c}}_\varepsilon + \beta \end{aligned} \tag{3.23}$$

$$\begin{aligned} \mathcal{T}(\bar{\mathbf{c}}) &= \mathcal{T}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\bar{\mathbf{c}}_\varepsilon\right) = \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\mathcal{T}(\bar{\mathbf{c}}_\varepsilon) \\ &= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}(\bar{\mathbf{c}}_\varepsilon + \beta) = \frac{n_\mathcal{O} + n_\mathcal{B}}{n}\beta + \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}}\frac{n_\varepsilon}{n}\bar{\mathbf{c}}_\varepsilon = \\ &= \bar{\mathbf{c}} + \beta \end{aligned} \tag{3.24}$$

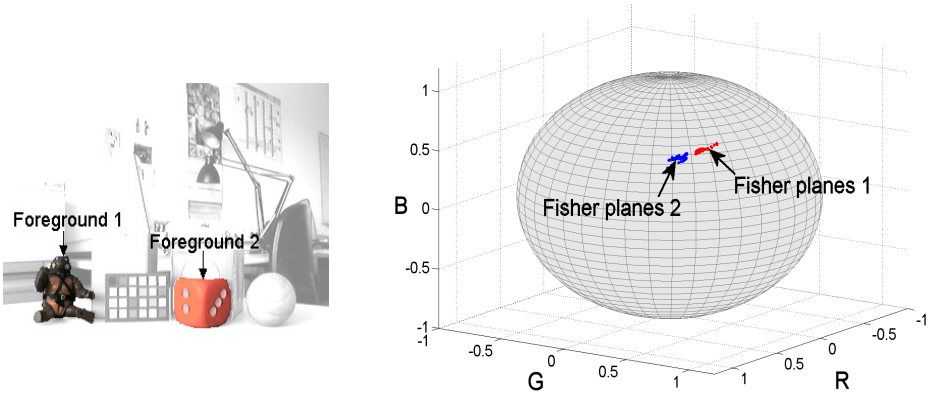
Figure 3.5: **Sample images of a illumination varying sequence.** Three distinct frames of the sequence used to analyze the performance of the Fisher colorspace in front of illumination changes. The frames correspond to the images of a still scenario (illuminated by natural light) acquired at different times of the day.

In a similar manner we can recompute the equations for the scatter matrices:

$$\begin{aligned}
\mathcal{T}(\mathbf{S}_w) &= \mathcal{T}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \sum_{i=1}^{n_\varepsilon} (\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)(\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)^T\right) \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \sum_{i=1}^{n_\varepsilon} (\mathcal{T}(\mathbf{c}_{\varepsilon,i}) - \mathcal{T}(\bar{\mathbf{c}}_\varepsilon))(\mathcal{T}(\mathbf{c}_{\varepsilon,i}) - \mathcal{T}(\bar{\mathbf{c}}_\varepsilon))^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \sum_{i=1}^{n_\varepsilon} (\mathbf{c}_{\varepsilon,i} + \beta - \bar{\mathbf{c}}_\varepsilon - \beta)(\mathbf{c}_{\varepsilon,i} + \beta - \bar{\mathbf{c}}_\varepsilon - \beta)^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} \frac{n_\varepsilon}{n} \sum_{i=1}^{n_\varepsilon} (\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)(\mathbf{c}_{\varepsilon,i} - \bar{\mathbf{c}}_\varepsilon)^T \\
&= \mathbf{S}_w
\end{aligned} \tag{3.25}$$

$$\begin{aligned}
\mathcal{T}(\mathbf{S}_b) &= \mathcal{T}\left(\sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} n_\varepsilon (\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})(\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})^T\right) \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} n_\varepsilon (\mathcal{T}(\bar{\mathbf{c}}_\varepsilon) - \mathcal{T}(\bar{\mathbf{c}}))(\mathcal{T}(\bar{\mathbf{c}}_\varepsilon) - \mathcal{T}(\bar{\mathbf{c}}))^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} n_\varepsilon (\bar{\mathbf{c}}_\varepsilon + \beta - \bar{\mathbf{c}} - \beta)(\bar{\mathbf{c}}_\varepsilon + \beta - \bar{\mathbf{c}} - \beta)^T \\
&= \sum_{\varepsilon=\{\mathcal{O},\mathcal{B}\}} n_\varepsilon (\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})(\bar{\mathbf{c}}_\varepsilon - \bar{\mathbf{c}})^T \\
&= \mathbf{S}_b
\end{aligned} \tag{3.26}$$

Thus, the scatter matrices of both the original images and the images affected by a lighting shifting are equal. As a consequence, the objective functions used to compute the



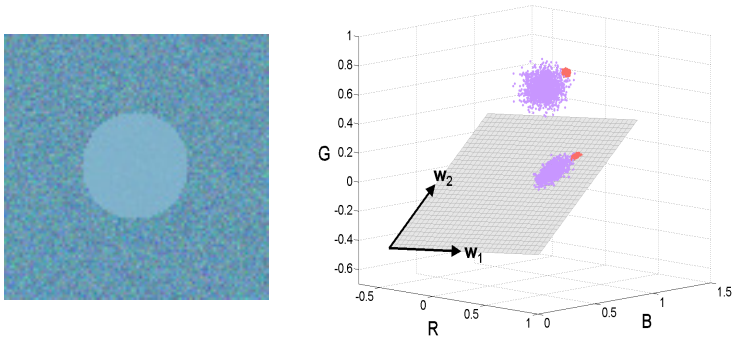
(a) (b)

Figure 3.6: **Fisher colorspace in front of illumination changes.** (a) Two different foregrounds for which the Fisher colorspace under different illumination conditions has been computed. (b) Representation of the Fisher colorspaces by their normal vectors. The surface of the sphere represents the whole space of possible Fisher plane configurations. However, the Fisher planes normals computed for each object are distributed on a small region onto the sphere ('Fisher planes i' are associated to 'Foreground i' in Fig. 3.6a). This proves that the Fisher colorspace presents some robustness in front of illumination changes.

Fisher plane are also equal, $\mathcal{T}(J(\mathbf{W})) = J(\mathbf{W})$, which means that the Fisher plane is also invariant to a illumination shifting effect. ■

The invariance of the Fisher plane to illumination changes is demonstrated in the following experiment, where the non parametric LDA analysis is applied to an image sequence of a still scenario illuminated by natural lighting that changes smoothly. The scene has been illuminated during a whole day, acquiring one shot per minute. Figure 3.5 shows three representative frames of the sequence.

Several foreground objects have been selected, and for each of them, the Fisher plane has been computed throughout the whole sequence. The results show that the Fisher planes (represented by their normal vectors) form separate clusters for every individual target, and the variance in each cluster is relatively small, proving that the Fisher colorspace is quite invariant to illumination changes. In Fig. 3.6b we depict the distribution of the normals to the Fisher plane for two different targets (indicated in Fig. 3.6a). The unitary sphere represents the space of all possible configurations of the normal to the Fisher plane. Observe how for each target, the Fisher plane distributions just occupy a small region onto the configuration space.



(a) (b)

Figure 3.7: **Performance of the Fisher colorspace for camouflaging targets.** (a) Experimental synthetic image: the target (central circle) has a similar appearance as the background of the image. (b) The Fisher colorspace provides a projection plane, where the target color distribution (red color points) can still be separated from the background color distribution (blue color points).

Nevertheless, in the real performance of the tracking system, the movement of the target through the scene may cause new objects to appear in the image. Moreover, object appearance may be affected by more complex illumination artifacts than the experienced in the previous example, such as specularities and interreflexions, and even, the object appearance might unexpectedly change. For all these reasons, in real applications the configuration of the Fisher plane may suffer noticeable modifications and will need to be adapted.

3.2.5 Fisher *versus* other colorspaces

In order to finalize this section dedicated to the selection of the appropriate colorspace for tracking tasks, we will proceed to compare the performance of the proposed Fisher colorspace, versus other commonly used colorspaces. In order to make a fair comparison we will consider only those colorspaces defined by two variables, such as the combination of two components of the normalized *RGB* colorspace (namely *rg*, *rb* and *gb*), the combination of two components of the normalized *XYZ* colorspace (*xy*, *xz*, and *yz*), and the two component combination of the *HSV* colorspace.

At this point, a natural question that might arise is why we do not use the complete representation of the colorspaces, i.e, the three components of the *RGB*, *XYZ* or *HSV* colorspaces, since it is known from the pattern recognition theory that when the dimensionality of the data is re-

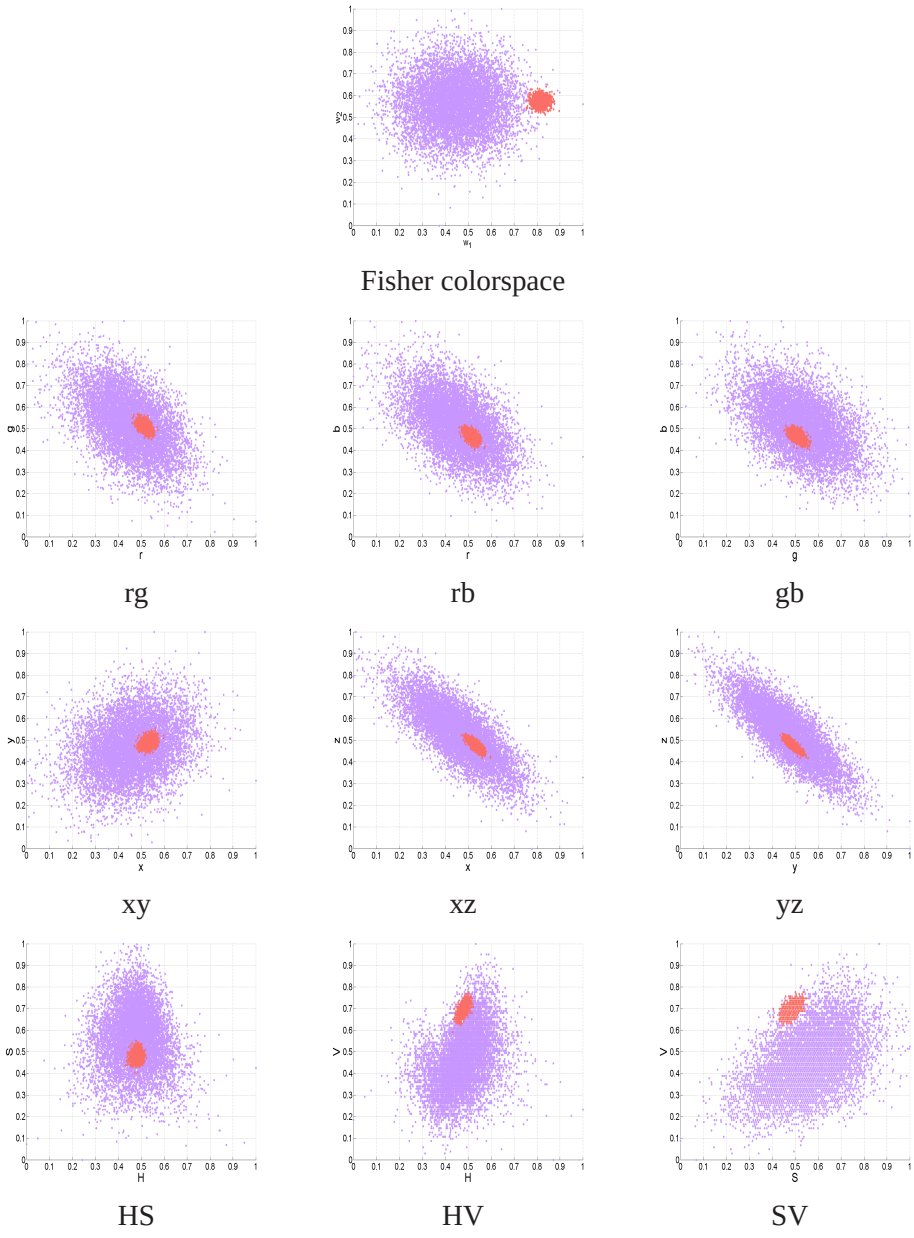


Figure 3.8: **Representation of the foreground and background of Fig. 3.7 on different colorspaces.** Observe that the representation on the Fisher colorspace is the one that provides a better foreground/background separation, making this colorspace appropriate for tracking tasks.

duced, relevant information might be lost. One reason why we represent the data by projecting from the 3D colorspace to a 2D colorspace, is that this projection is similar to a normalization process, and as it is observed in (36), normalization procedures reduce the dependencies of the perceived colors to the illuminant color or to the geometry of the light sources. Furthermore, by reducing the color representation from 3D to 2D, the complexity of the subsequent operations may be considerably simplified: fitting a parametric model to the color distributions represented on a bidimensional space is much more simple than in 3D space, and the dynamics to predict the movement of the color distributions on the plane may be more easily learned than when the color distributions move freely in the 3D space.

Having made this clarification we proceed with the comparison of different colorspaces. We begin by showing a naive example, where all but the Fisher colorspace are not capable to properly separate the object from the background. Figure 3.7a shows the test image, where the central circle is the target and the rest of the image represents the background. Observe that the appearance of both classes is quite similar. In these circumstances where the target is in some degree camouflaged with the background, the Fisher colorspace clearly performs better than other colorspaces. Figure 3.7b shows the color distributions of all image points, represented into the RGB colorspace. Blue dots correspond to the background points, and red dots are the target color points. Note that both classes are in close contact. In spite of this, the projection of the colorpoints onto the Fisher plane does not overlap the target and background classes. On the other hand, when we represent the points using the other colorspaces, the two classes are greatly overlapped, which will cause difficulties in future tracking tasks. Observe this in Fig. 3.8, where the representations of the foreground and background classes are depicted for each one of the colorspaces considered in the comparison.

A more precise comparison is obtained by computing the distance between the representation of the target and the background for each one of the colorspaces. In order to eliminate a possible effect of scale when computing the distance, the data represented in each particular colorspace is normalized. Subsequently, given the set of color points belonging to the object $\mathbf{Y}_{\mathcal{O}} = \{\mathbf{y}_{\mathcal{O},1}, \ldots, \mathbf{y}_{\mathcal{O},n_{\mathcal{O}}}\}$ and the set of background color points $\mathbf{Y}_{\mathcal{B}} = \{\mathbf{y}_{\mathcal{B},1}, \ldots, \mathbf{y}_{\mathcal{B},n_{\mathcal{B}}}\}$ points, the measure of distance between both sets is computed through the following metric:

$$dist(\mathbf{Y}_{\mathcal{O}}, \mathbf{Y}_{\mathcal{B}}) = \frac{\frac{1}{n_{\mathcal{O}}} \sum_{i=1}^{n_{\mathcal{O}}} \frac{1}{k} \sum_{j=1}^{k} \|\mathbf{y}_{\mathcal{O},i} - \mathcal{NN}^{j}_{\mathcal{B}}(\mathbf{y}_{\mathcal{O},i})\|^2}{|\det(\mathbf{S}_w)|} \tag{3.27}$$

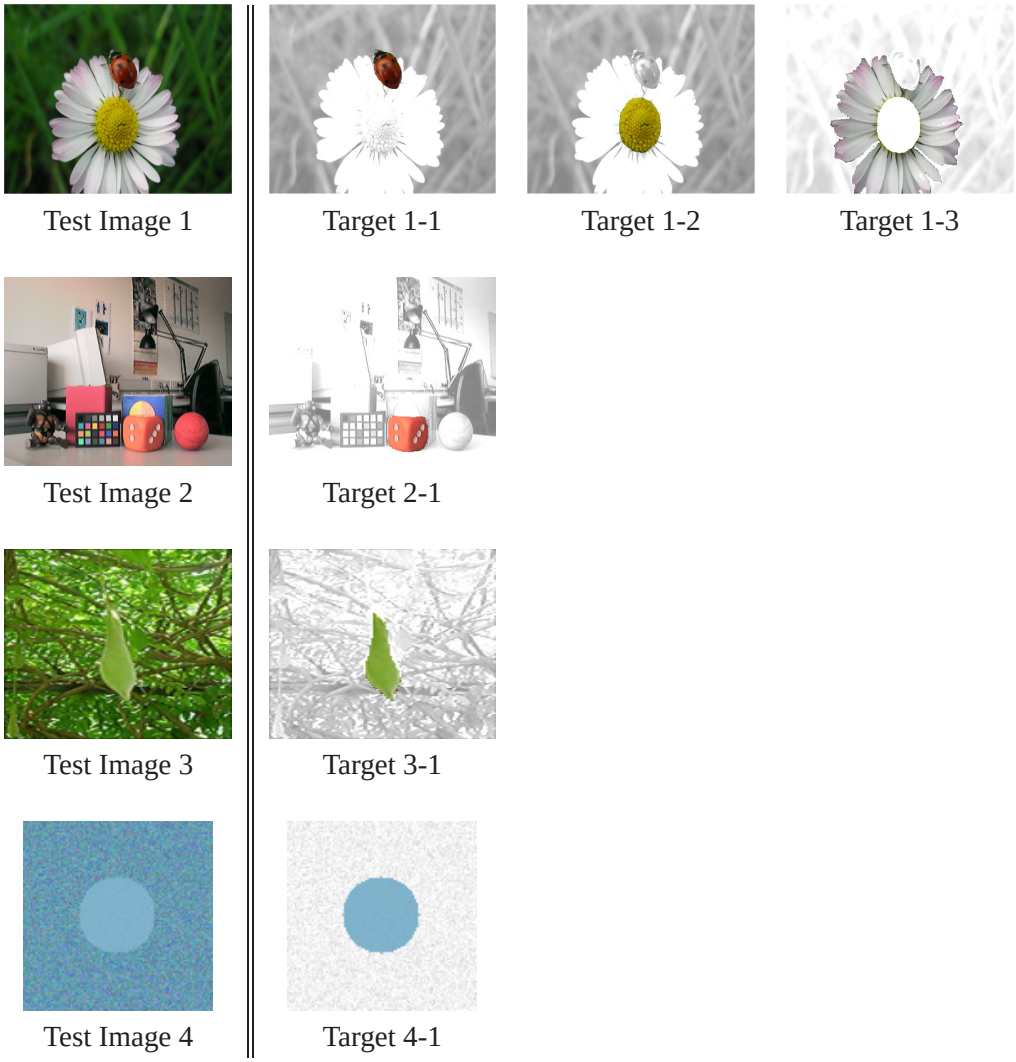


Figure 3.9: **Test images used to compare the performance of different colorspaces.** In each row, are shown the test image and various targets for which the colorspaces are evaluated.

where $\mathcal{NN}^j_{\mathcal{B}}(\mathbf{y}_{\mathcal{O},i})$ is the j-th nearest neighbor in the set $\mathbf{Y}_{\mathcal{B}}$ to a point $\mathbf{y}_{\mathcal{O},i}$, $\|\cdot\|$ is the Euclidean norm, and $|\cdot|$ the absolute value function. $\mathbf{S}_w$ is the *within class scatter matrix* previously defined (Eq. 3.10). Note that with this metric equation we define the distance between the foreground and background sets in the same terms as it was done with the Fisher objective function (Eq. 3.9). That is, the metric is computed by considering the distance of each foreground point $\mathbf{y}_{\mathcal{O},i}$, $i = 1, \ldots, n_{\mathcal{O}}$ with respect to its k-*nearest neighbors* in the background set of points $\mathbf{Y}_{\mathcal{B}}$. Since we are also searching for a colorspace minimizing the variance of each

class, the distance measure is divided by the determinant of the *within class scatter matrix* $\mathbf{S}_w$.

According to this metric definition, we have computed the foreground background distance for the test images and corresponding targets depicted in Fig. 3.9. Distance results for each one of the experiments are shown in Table 3.1. Observe that in most of the experiments, the Fisher colorspace proposed in this thesis is the colorspace providing a color representation of the object and background with maximal separation.

	Target 1-1	**Target 1-2**	**Target 1-3**	**Target 2-1**	**Target 3-1**	**Target 4-1**
Fisher	429.1	2445.5	202.1	133.8	9.9	74.2
rg	464.3	214.8	8.2	63.9	2.7	4.8
rb	115.5	219.1	4.0	61.6	1.8	4.9
gb	117.5	183.7	4.5	64.5	2.9	4.8
xy	141.3	275.0	2.8	54.3	1.7	4.5
xz	115.1	303.7	3.5	70.9	1.9	6.8
yz	87.7	274.1	4.7	88.0	4.2	7.9
HS	5.8	5.6	147.7	4.5	3.1	5.1
HV	5.4	1.3	156.1	0.3	5.1	16.4
SV	1.0	16.2	110.3	1.4	5.1	32.8

Table 3.1: **Foreground-Background distances in various colorspaces.** Each column represents a different experiment where the distance between the target and the background of the image is evaluated according to Eq. 3.27. The label of each column indicates the corresponding target in Fig. 3.9, and the rows consider different colorspaces. Note that the Fisher colorspace proposed in this thesis provides the best results (maximal foreground-background distance) in almost all the experiments.

3.3 Color distribution representation

After having selected the colorspace, the next step is to chose a model for representing the color distribution of the object and background when *RGB* colorpoints are projected onto the Fisher plane. One possible choice, consists of representing the color of the object by non-parametric histograms. This technique has been extensively used in the computer vision community, especially in location and tracking applications (13; 56; 91; 100; 119; 123; 145). Although color histograms have been demonstrated to be an effective and easy to implement tool for approximating monochromatic color distributions, when the object to be modeled contains regions with different colors, the number of pixels representing each color may be relatively low, and a histogram representation might not suffice.

In these circumstances the approximation of the color distributions by a *Mixture of Gaussians* (MoG) model, represents a more effective approach (54; 109; 144). Using this model, the conditional probability for a pixel $\mathbf{f} \in \mathbb{R}_{2\times 1}$ (represented in the Fisher colorspace) belonging to a multi-colored object $\mathcal{O}$ is expressed as a sum of $m_{\mathcal{O}}$ Gaussian components:

$$p(\mathbf{f}|\mathcal{O}) = \sum_{j=1}^{m_{\mathcal{O}}} p(\mathbf{f}|\mathcal{O}, j)P(\mathcal{O}, j) \tag{3.28}$$

where $P(\mathcal{O}, j)$ corresponds to the a priori probability that pixel $\mathbf{f}$ was generated by the j-th Gaussian component of the object color distribution, and where $\sum_{j=1}^{m_{\mathcal{O}}} P(\mathcal{O}, j) = 1$. Each component is a Gaussian distribution in the Fisher colorspace, with mean $\boldsymbol{\mu}_{\mathcal{O},j}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathcal{O},j}$, i.e,

$$p(\mathbf{f}|\mathcal{O}, j) = \frac{1}{\sqrt{\det(2\pi\boldsymbol{\Sigma}_{\mathcal{O},j})}} \exp\left[-\frac{1}{2}(\mathbf{f} - \boldsymbol{\mu}_{\mathcal{O},j})^T \boldsymbol{\Sigma}_{\mathcal{O},j}^{-1}(\mathbf{f} - \boldsymbol{\mu}_{\mathcal{O},j})\right] \tag{3.29}$$

Similarly, the color distribution of the background points may be represented by a mixture of $m_{\mathcal{B}}$ Gaussian components.

3.3.1 Object segmentation using color

In order to track and segment the object out from the background, we may wish to compute the a posteriori probability that each image pixel $\mathbf{f}$ belongs to the object $\mathcal{O}$. Given the density estimates for both the object $\mathcal{O}$ and the background $\mathcal{B}$, this a posteriori probability is given by the Bayes rule:

$$p(\mathcal{O}|\mathbf{f}) = \frac{p(\mathbf{f}|\mathcal{O})P(\mathcal{O})}{p(\mathbf{f}|\mathcal{O})P(\mathcal{O}) + p(\mathbf{f}|\mathcal{B})P(\mathcal{B})} \tag{3.30}$$

where $P(\mathcal{O})$ and $P(\mathcal{B})$ are the prior probabilities of the object and background classes respectively.

Fig. 3.10b shows an example of the *Mixture of Gaussians* model fitted to the foreground and background color distributions (in the Fisher colorspace) of the image depicted in Fig 3.10a, where the ladybird is selected as foreground. In this example, the foreground color distribution is approximated by 5 Gaussian components, while the background distribution is approximated with 4 components. The selection of the optimal number of components that best approximate the data is not a trivial issue, and it will be discussed in the next subsection.

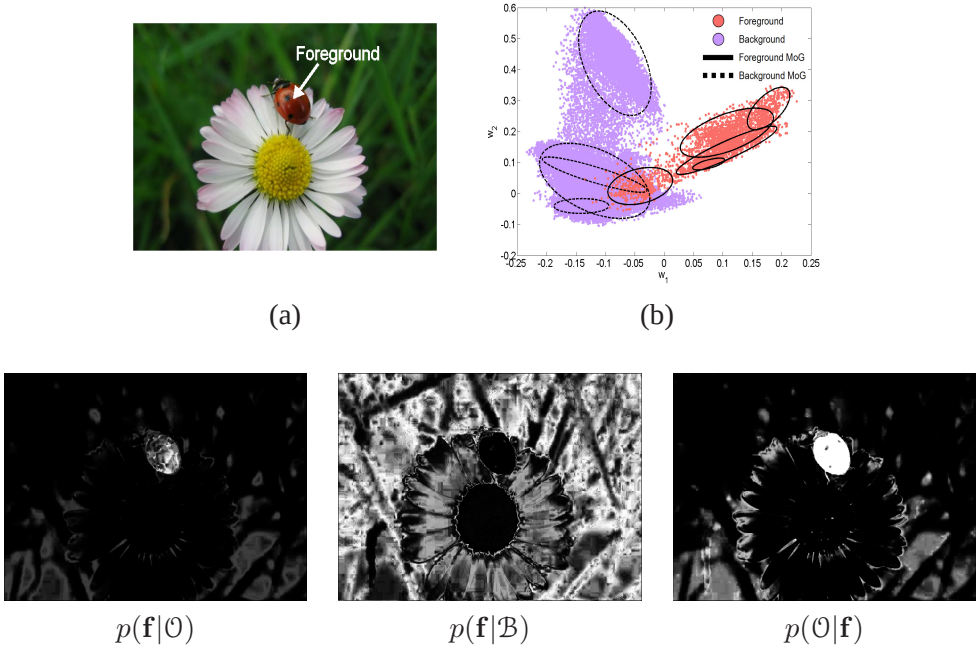


Figure 3.10: **Foreground and background color distribution parameterization.** Upper row: (a) Original image, indicating the foreground (ladybird). (b) Foreground and background color distributions and the corresponding Mixture of Gaussians models in the Fisher colorspace. Lower row: Probability maps involved in the computation of the a posteriori probability distribution of the object. $p(\mathbf{f}|\mathcal{O})$ and $p(\mathbf{f}|\mathcal{B})$ are the a priori probability maps of the object and background, respectively. Based on the Bayes rule, these conditional densities permit to compute the a posteriori probability map of the object class. Brighter points correspond to more likely pixels.

Once the Mixture of Gaussians model is adjusted to the data, the a posteriori probability map associated to the foreground class, may be computed by first calculating the a priori conditional densities $p(\mathbf{f}|\mathcal{O})$ and $p(\mathbf{f}|\mathcal{B})$. Subsequently, using the Bayes rule, these densities allow to compute the a posteriori probability map $p(\mathcal{O}|\mathbf{f})$ (Fig. 3.10, lower row). In this image, brighter points correspond to pixels that more likely will belong to the object.

3.3.2 Parameter estimation

The computation of Eq. 3.28 requires the estimate of the set of parameters of the Mixture of Gaussians model for both the foreground and background color distributions (necessary to determine $p(\mathbf{f}|\mathcal{O})$ and $p(\mathbf{f}|\mathcal{B})$). Furthermore the prior probabilities $p(\mathcal{O})$ and $p(\mathcal{B})$ need to be approximated, and are usually initialized to the expected area ratios of the foreground

and background classes in the image. In order to estimate the parameters of the Gaussian distributions, the *Expectation-Maximization* (EM) algorithm (15; 29) is the tool commonly used in the computer vision literature (see for instance (54; 109; 144)). Next, we give a broad outline of the EM algorithm.

Let $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_n]$ be a set of d-dimensional data points. The goal of the EM algorithm is to determine the parameters of a Mixture of Gaussians model that fits the data. These parameters are the prior probabilities $P(j)$, the set of means $\boldsymbol{\mu}_j$ and covariance matrices $\boldsymbol{\Sigma}_j$, for $j = 1, \ldots, m$, where m is the number of Gaussian components, which needs to be specified a priori. The EM algorithm is based on an initialization stage followed by an iterative process between the *expectation* and *maximization* stages:

Parameter Initialization: The position of the means $\boldsymbol{\mu}_j$, the value of the covariance matrices $\boldsymbol{\Sigma}_j$ and the priors $P(j)$ need to be initialized. Bishop (15) suggests an initialization strategy where the centers $\boldsymbol{\mu}_j$ are placed on a random subset of the data. $\boldsymbol{\Sigma}_j$ are initialized to matrices $\sigma_j \mathbb{I}$, where $\mathbb{I}$ is the identity matrix and σ_j is the distance between $\boldsymbol{\mu}_j$ and the closest center. Finally all the priors $P(j)$ are initialized to $1/m$.

Expectation: The *expectation* stage consists in evaluating the a posteriori probabilities $p(j|\mathbf{f}_i)$ for each component of the Gaussian mixture. $p(j|\mathbf{f}_i)$ represents the probabitility that the data point $\mathbf{f}_i$ was generated by the j-th Gaussian component. It may be computed by using the Bayes rule, according to:

$$p(j|\mathbf{f}_i) = \frac{p(\mathbf{f}_i|j)P(j)}{p(\mathbf{f}_i)} = \frac{p(\mathbf{f}_i|j)P(j)}{\sum_{j=1}^{m} p(\mathbf{f}_i|j)P(j)} \tag{3.31}$$

Let us call the sum of all these probabilities, the *weight* of the j-th component:

$$w_j = \sum_{i=1}^{n} p(j|\mathbf{f}_i) \tag{3.32}$$

Maximization: In this stage, the components of the Mixture of Gaussians are updated according to the following expressions:

$$P(j)^{new} = \frac{w_j}{n} \tag{3.33}$$

$$\boldsymbol{\mu}_j^{new} = \frac{1}{w_j} \sum_{i=1}^{n} \mathbf{f}_i \, p(j|\mathbf{f}_i) \tag{3.34}$$

$$\boldsymbol{\Sigma}_j^{new} = \frac{1}{w_j} \sum_{i=1}^{n} \left(\mathbf{f}_i - \boldsymbol{\mu}_j^{new}\right) \left(\mathbf{f}_i - \boldsymbol{\mu}_j^{new}\right)^T p(j|\mathbf{f}_i) \tag{3.35}$$

The stages *expectation* and *maximization* are iterated until convergence.

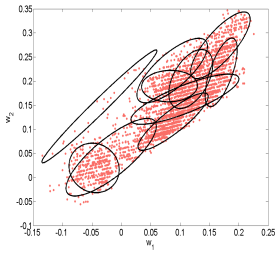
However, as it is observed by Figueiredo and Jain (35), the standard formulation of the EM algorithm has several drawbacks, such as a high sensitivity to the parameter initialization phase and that the number of Gaussian components needs to be specified a priori and it is not automatically selected.

In this thesis we will estimate the parameters of the Mixture of Gaussians based on the approach proposed in (35), which is basically a variant of the EM algorithm with some modifications allowing to overcome the limitations of the standard EM algorithm. The method starts with a large number of mixture components and successively annihilates components for which the weight defined in Eq. 3.32 is small. By starting with an initial number of Gaussian components much larger than the optimal number, the dependence with the initial parameters is reduced. The problem with the parameter initialization arises when there are too many components in a region of the space, and too few in another. As it is noted in (35), in these circumstances the EM algorithm is unable to move components across low-likelihood regions, resulting in local maxima solutions. Nevertheless, by starting with a high number of components and removing the unnecessary ones, this problem is avoided. The *expectation*, *maximization* and *annihilation* stages are iterated until the convergence of a cost function, based on a *Minimum Message Lenght* criterion (133).

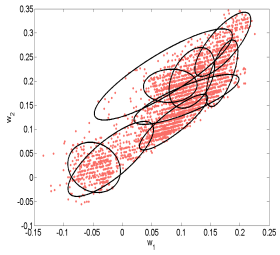
Note that using this approach, both the problem of selecting the optimal number of components as the initialization dependence problem are simultaneously solved. Fig. 3.11 shows an example of the fitting procedure, for the color distribution of the ladybird depicted in Fig. 3.10a. The process is initialized by considering a large number of components ($m = 10$), in such a way that it is guaranteed that all the regions of the space containing data are covered by a Gaussian. Subsequently, the *expectation* and *maximization* stages are iterated seeking for the minimum of the *message length* cost function (the *message length* value associated to each configuration is indicated below the plots). During the fitting process, those Gaussian components not supported by the data are annihilated. Observe how removing some components, the cost function is minimized, reaching the minimum value for $m = 5$ components, which is the configuration chosen to approximate the color distribution.

3.4 Contour representation

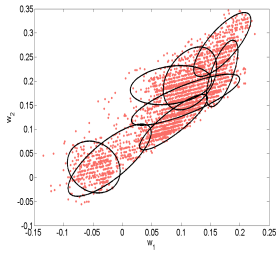
Since color segmentation usually only gives a rough estimation about the object location, we include the contour feature in the object representation. The use of geometric cues, although



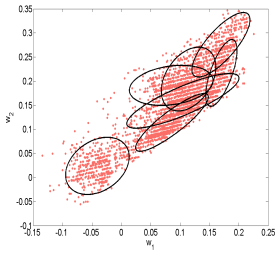
m=10, Mess.Lenght=-1.3071



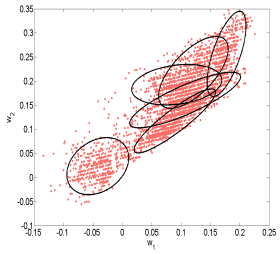
m=9, Mess.Lenght=-1.3074



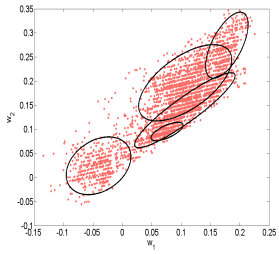
m=8, Mess.Lenght=-1.3066



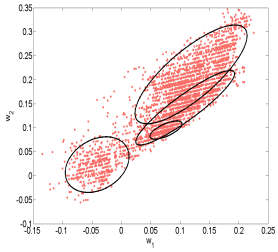
m=7, Mess.Lenght=-1.3060



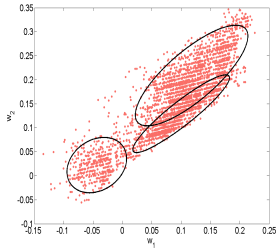
m=6, Mess.Lenght=-1.3046



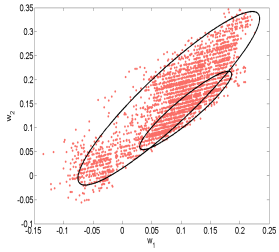
m=5, Mess.Lenght=-1.3165



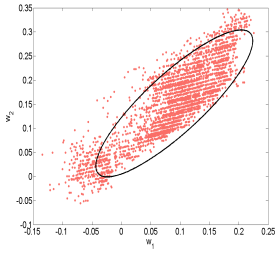
m=4, Mess.Lenght=-1.3098



m=3, Mess.Lenght=-1.2961



m=2, Mess.Lenght=-1.2551



m=1, Mess.Lenght=-1.2009

Figure 3.11: **Fitting a Mixture of Gaussians.** Automatic model selection for the color distribution of the ladybird (Fig. 3.10a) using the method proposed in (35). Initially a large number of components is selected ($m = 10$), but the process automatically annihilates the Gaussian components not supported by the data. Below each plot we indicate the value of the cost function (Message lenght $\times 1.0e-4$). The configuration that minimizes the cost function, and thus, best approximates the data, corresponds to $m = 5$.

being more sensitive to clutter than appearance cues, allows to infer the position of the object with high accuracy.

The contour of the object is modeled as a curve, which is discretized and represented by a set of n_c points in the image:

$$\mathbf{R} = \left[(u_1, v_1)^T, \ldots, (u_{n_c}, v_{n_c})^T\right]^T \tag{3.36}$$

where (u_i, v_i) are the 2D image coordinates of the i-th point of the curve.

3.4.1 Adjusting the curve to the object countour

In order to adjust the curve describing the contour, to the object boundary, even when the object suffers from non rigid deformations, we will make use of the *snake* (or deformable curve) formulation (59).

A snake is a curve $\mathbf{r}(s) = [u(s), v(s)]$, $s \in [0, 1]$, which moves through the image. In the traditional snake formulation, the problem of adjusting a snake curve to the boundary of an object can be viewed as a force balance equation:

$$\mathcal{F}_{int}(\mathbf{r}(s)) + \mathcal{F}_{ext}(\mathbf{r}(s)) = 0 \tag{3.37}$$

where $\mathcal{F}_{int}(\mathbf{r}(s)) = \alpha\frac{\partial^2\mathbf{r}(s)}{\partial s^2} + \beta\frac{\partial^4\mathbf{r}(s)}{\partial s^4}$ are the internal forces that control the bending and stretching of the snake (α and β are the elasticity and rigidity parameters, respectively). $\mathcal{F}_{ext}(\mathbf{r}(s))$ are the external forces that pull the curve toward the edge image features. In the literature, there exist several definitions for this external function. For instance, given a gray-level image $\mathbf{I}(u, v)$, a typical external force used to pull the deformable contour toward the edges is the Laplacian of the image, i.e, $\mathcal{F}_{ext} = |\nabla^2\mathbf{I}(u, v)|$, where ∇^2 is the Laplacian operator. In particular, in this thesis we will use the the *Gradient Vector Flow* (GVF) external force proposed in (143), because it exhibits a larger capture range and better convergence performance in boundary concavities than other methods. This external force, is computed as a diffusion of the gradient vectors of a binary edge map from the image. Specifically, we generate the edge map, using the *Canny* algorithm (21).

Equation 3.37 may be solved by writing the snake as a function of both space and time, i.e, $\mathbf{r}(s) \rightarrow \mathbf{r}(s, t)$ (we will write $\mathbf{r}^t$) and iterating over the following expression:

$$\frac{\mathbf{r}^t - \mathbf{r}^{t-1}}{\triangle t} = \alpha\frac{\partial^2\mathbf{r}^{t-1}}{\partial s^2} + \beta\frac{\partial^4\mathbf{r}^{t-1}}{\partial s^4} + \mathcal{F}_{ext}(\mathbf{r}^{t-1}) \tag{3.38}$$

(a) (b) (c)

Figure 3.12: **Fitting a deformable curve to an object contour.** (a) Original image and initial configuration of the contour. (b) Fitting process. Observe that using the *Gradient Vector Flow* (143) external force, the snake can be fitted into boundary concavities. (c) Final configuration of the snake.

where $\triangle t$ is the time step for each iteration. When the solution stabilizes ($\mathbf{r}^{t-1} = \mathbf{r}^t$), Eq. 3.37 is satisfied.

For the numerical implementation we approximate the derivatives with finite differences, and discretize the curve $\mathbf{r}(s,t)$ with n_c points, so that the previous gradient descent method can be rewritten as:

$$\mathbf{R}^t = (\mathbb{I} - \triangle t\mathbf{Q})^{-1}(\mathbf{R}^{t-1} + \triangle t\mathcal{F}_{ext}(\mathbf{R}^{t-1})) \tag{3.39}$$

where $\mathbf{R}$ is the matrix defined in Eq. 3.36 containing the coordinates of the n_c points of the contour, $\mathbf{Q}$ is a $n_c \times n_c$ pentadiagonal matrix including the α and β parameters, and $\mathbb{I}$ is the $n_c \times n_c$ identity matrix. Iterating over Eq. 3.39 the snake is locally fitted to the edges of the image, only governed by its internal and external forces.

Fig. 3.12 depicts the process of fitting a deformable curve to the contour of an object. This particular example shows the ability of the GVF snake to move into boundary concavities.

3.4.2 Robustness to clutter

One of the limitations of the contour based tracking algorithms, is that they are prone to be sensitive to noisy edges that may be caused by a cluttered background. These false edges, might distract the snake during the fitting procedure, unless it is initialized close to the true object boundary. The example of Fig. 3.12 did not suffer from this drawback since the image had no background. However, in the example of Fig. 3.13, when attempting to fit a deformable

Initial configuration | Non constrained snake | Affine constrained snake

Figure 3.13: **Fitting a deformable curve to an object contour in cluttered images.** (a) Initial configuration of the contour in an image with a high level of clutter. The shell of the snail is the desired target. (b) The presence of clutter, generates false edges which distract the snake to converge to the true object boundary. (c) If the snake is enforced to evolve under an affine constraint, the correct silhouette of the tracked object may be maintained. However, this does not guarantee the convergence to the true object boundary.

contour to the shell of the snail, the clutter and noise of the image causes confusion to the snake, and it converges to false edges. Observe that even in the case that the contour is initialized close to the true boundary (Fig. 3.13a), the snake algorithm fails when attempting to fit the contour of the shell (Fig. 3.13b).

In order to reduce the influence of the distractors, some constraints might be enforced to the movement of the snake. For instance, an affine constraint could be introduced into the snake movement by considering the following equations:

$$\mathbf{R}_H^t = (\mathbb{I} - \triangle t\mathbf{Q})^{-1}(\mathbf{R}_H^{t-1} + \triangle t\mathcal{F}_{ext}(\mathbf{R}_H^{t-1})) \tag{3.40}$$

$$\mathbf{R}_H^t = \mathbf{R}_H^{t-1}\mathbf{H} \tag{3.41}$$

$$\text{where} \quad \mathbf{H} = \begin{bmatrix} a_{11} & a_{12} & t_1 \\ a_{21} & a_{22} & t_2 \\ 0 & 0 & 1 \end{bmatrix}$$

Note that Eq. 3.40 is the same as Eq. 3.39, except that the matrix $\mathbf{R}_H$ contains the homogeneous coordinates of the n_c points of the curve, i.e, $\mathbf{R}_H = \left[(u_1, v_1, 1)^T, \ldots, (u_{n_c}, v_{n_c}, 1)^T\right]^T$. Combining Equations 3.40 and 3.41, we obtain the following iterative procedure for the affine snake deformation:

> Given the contour $\mathbf{R}_H^{t-1}$ at time $t-1$, the matrix of parameters $\mathbf{Q}$ and the time step $\triangle t$, the configuration of the contour is updated through the following steps:

1. Estimate the affine matrix $\mathbf{H} = (\mathbf{J}^T\mathbf{J})^{-1}(\mathbf{R}_H^{t-1} + \mathcal{F}_{ext}(\mathbf{R}_H^{t-1}))$
   where $\mathbf{J} = \mathbf{R}_H^{t-1} - \triangle t\mathbf{Q}\mathbf{R}_H^{t-1}$.
2. Normalize $\mathbf{H}$ with respect to the component $H(3,3)$.
   Set $H(3,1) = H(3,2) = 0$.
3. Update the contour points, i.e, $\mathbf{R}_H^t = \mathbf{R}_H^{t-1}\mathbf{H}$.

Steps 1-3 are iterated until the convergence of $\mathbf{R}_H^t$ and $\mathbf{R}_H^{t-1}$.

These kind of constraints improve the performance of the contour adjustment in the presence of clutter. However, they might not be enough to obtain a robust tracking if the level of noise is relatively high. For instance, in the example of Fig. 3.13 previously introduced, we can observe that even though the affine constrains allow to maintain the correct silhouette of the object, the contour still does not converge to the true object boundary.

As it will be discussed in next chapters, the integration of other cues, such as color, may be used to remove most of the noisy edges, which may guarantee the adjustment of the deformable curve to the true object boundary.

3.5 Bounding box representation

The last object cue that will be considered refers to a module providing a rough estimation of the object position, which will be used as initialization for other features. Specifically, in order to adjust the colorspace feature introduced earlier, we will need an approximation of the region in the image where the target is expected to be. This region does not need to be estimated precisely, and will be represented as a rectangular bounding box surrounding the object, with a sufficient gap to ensure that it contains the object.

One particularity of this feature, is that its estimation will be performed using the Kalman filter instead of using particle filters. Furthermore, the correction term of the Kalman filter, will have low importance, and the term with major importance in the estimation, will be the prediction. Again, in the following chapters we will extend these ideas.

3.6 Summary

The selection of appropriate features in order to represent the object is a key initial issue in any computer vision application, and specially for tracking purposes. These features, must allow to characterize the target, and discriminate it from the rest of objects in the image.

In the present chapter we have described the individual features that will be used to represent the object. At this point, we have not considered the relation among features; this will be covered in Chapter 5, when describing the feature integration scheme.

In particular, we have described the object by appearance and geometric cues. Special attention should be paid to the definition of a colorspace dependent on the object appearance. This colorspace, which we call *Fisher colorspace*, has the capability of maximizing the distance between the representation of the target and background colorpoints, which is an important property for any color based tracking system. The further are the object and background representations, the easiest will be the segmentation, and therefore, the tracking process.

We have described the parameterization of the color distributions by *Mixture of Gaussians* models, and the initialization of its parameters using a variant of the *Expectation Maximization* algorithm.

In order to obtain a more precise information about the target position, we have also considered the representation of the object through geometric cues. An accurate estimate of the object location may be obtained considering the contour feature, described by the points that discretize a deformable curve (or snake). Furthermore, for initializing the colorspace feature, the bounding box of the target will be used to provide a coarse estimate about the object location.

Chapter 4

Multiple cues integration for tracking tasks: a review

Tracking and figure-ground segmentation of image sequences is a topic of great interest in a wide variety of computer vision applications, extending from video compression to mobile robot navigation. It has been observed that the simultaneous use of redundant and complementary cues for describing the target, increases noticeably the robustness of the system to non-stationary environments, suffering from abrupt changes of illumination, cluttered backgrounds and non-linear dynamics of the object movement/deformation. Although several surveys are available for the general case of sensor fusion, these kind of reviews do not exist for the specific case of unifying different types of data present in images, such as color, contours or motion.

In this chapter, we survey previous work on multiple cue integration techniques for tracking and figure-ground segmentation tasks. We review, classify and analyze what we feel are the most relevant approaches in the field over the past decade, discussing its relationship to the framework presented in this thesis, which will be covered in detail in Chapter 5.

4.1 Introduction

Visual tracking and segmentation of the foreground objects out from the background in video sequences are important initial tasks for a high number of computer vision applications, such as object recognition, assembly in industrial tasks, mobile robot navigation, automatic video editing and summarization, or video coding, where for instance codecs based on MPEG-4 require different objects to be represented separately.

It has been observed that the simultaneous use of complementary and redundant cues when

representing the object (color, shape, geometry, motion, etc.), can significantly enhance the figure-ground segmentation results. For example, based on the color distribution of the object, one could increase the robustness of a contour tracking in a highly cluttered environment. Conversely, the integration of shape and color cues could allow better tracking in the presence of color distracters.

During the last decade, the number of papers dealing with the fusion of visual object cues in the design of computer vision applications has grown significantly. Nevertheless, is does not exist a survey paper giving an overview of the individual papers and structuring them into groups depending on similar characteristics. These kind of surveys and taxonomies are only available for the general case of multisensor fusion (9; 62; 74), which usually deals with the integration of information provided by different sources (for instance integration of visual and tactile sensing to identify and locate objects from among a group of known objects, such is described in (44)). As a consequence, we think that a survey dealing exclusively with the integration of visual modules for figure-ground segmentation (and tracking) tasks is of great interests for the computer vision community. This is the main goal of the present chapter, to propose a taxonomy and review the state of the art of such techniques and relate them with the cue integration framework presented in this dissertation, which will be discussed in detail in the following chapter.

In order to discriminate the object of interest from the background, the target can be characterized by a set of different visual features. The features most commonly used in the literature are: motion, color, contrast, texture, appearance and shape (the contour if we work with 2D data, or the target surface if we work with 3D data). The purpose of the fusion techniques is to properly combine the information of each of these features in order to infer the regions in consecutive images having higher probability of belonging to the object.

Some desirable features of the fusion scheme (that will be taken into account later in the chapter when analyzing different fusion algorithms) are the following:

- Robustness: In order to design a system robust to challenging environments, with cluttered backgrounds that might be confused with the target, changes of illumination and non-linear dynamics, the fusion scheme needs to benefit from all the cues.

- Adaptability: Some of the cues which usually are reliable may degrade performance, sometimes abruptly, in certain situations. In these circumstances, it is necessary a fusion

scheme allowing for the automatic adaptability of the features. Adaptability is a key feature to provide robustness.

- Modularity/Complexity/Scalability: In order that the system complexity does not increase significantly when integrating additional object features, it is interesting to represent each cue by a separated module with the input and output signals clearly defined. This allows to scale the system and integrate additional features with a reduced cost and effort.

- Temporal consistency: Since we are dealing with video sequences, the fusion methodology needs to impose temporal consistency constrains in the segmentation results of consecutive frames.

- Prediction module: If the target and the clutter are indistinguishable in terms of their representations, the fusion scheme needs a prediction module in order to determine the object segmentation at least for a short period of time, by using the prior knowledge about the dynamics.

After this short introduction, the rest of the chapter is organized as follows. Section 4.2 describes the proposed taxonomy for multiple cue integration techniques tailored to tracking tasks. Section 4.3 reviews the state of the art of the fusion techniques and classifies them according to the proposed taxonomy. In Section 4.4, an analysis of different quality factors of the fusion approaches is performed, and finally, Section 4.5 summarizes this chapter, and establishes the bases to describe the cue integration framework proposed in this dissertation, which will be discussed in next chapter.

4.2 Classification of multiple cues integration techniques

In this section we propose a general classification of the techniques for integrating multiple visual modules in tracking tasks.

If we take into account the flow of information in the fusion scheme, we can classify the fusion techniques as it is done by Clark and Yuille in (23). Note that the classification scheme proposed by Clark and Yuille, refers to the general case of multiple sensor integration techniques. We use the same general classification for the specific case of multiple cues integration, for visual tracking tasks. Afterwards we propose a secondary classification and subdivide each

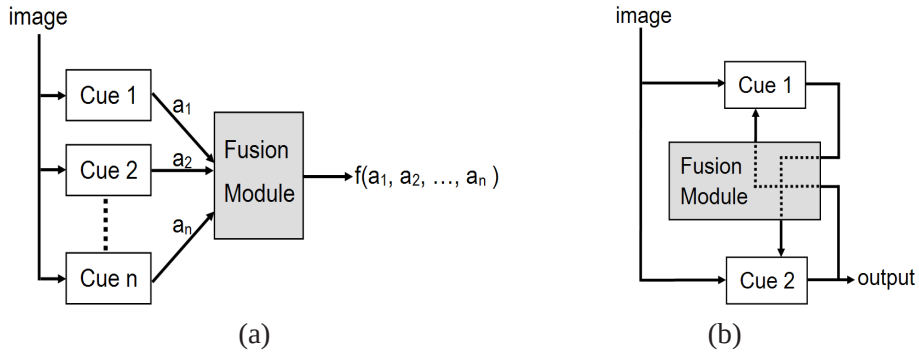


Figure 4.1: **Categorization of the fusion algorithms into (a) weakly coupled and (b) strongly coupled classes.**

group in smaller classes. In (23), two major classes of fusion methods are considered, namely the *weakly coupled* and *strongly coupled*. Into the former class, we would include those algorithms which outputs are independently combined, i.e. the operation of each algorithm is not affected by the fusion process (see Fig. 4.1a). Into the strongly coupled methods, the outputs of different modules are no longer independent and the operation of one algorithm is affected by the output of another module (see Fig. 4.1b).

Strongly coupled data fusion algorithms permit to represent dependencies between features, providing improved priors for each algorithm (each algorithm is assumed to estimate the state of a different cue). For instance, a usual method to evaluate the quality of an object contour hypothesis is based on the ratio of the number of pixels inside the contour with object color versus the number of pixels outside the contour with background color. This means that the contour feature is not independent of the color feature, and this dependence can be represented using the strongly coupled methods. However, these methods are prone to be sensitive to errors in a specific module, in such a way that an invalid output might be propagated through the fusing scheme and have a negative effect in the subsequent modules. Moreover, strongly coupled algorithms usually require the knowledge of feature interdependencies, and sometimes this is not a trivial issue.

On the other hand, weakly coupled algorithms allow the design of straightforward fusion methods, and the effect of a failure in one of the algorithms does not have repercussion in other modules. Nevertheless, this fusion methodology is unable to deal with complex systems containing highly correlated object features.

Furthermore, we deem interesting to propose a secondary subdivision of the weakly and strongly coupled methods based on the intrinsic methodology used when performing the fusion. If the primary classification was based on the flow of information between algorithms, this secondary classification has to do with the intrinsic mathematics involved when processing this information. From this point of view, and considering the study performed in previous works (1; 17; 72; 74), weakly coupled algorithms can be combined according to the methodologies of *weighted average*, *voting*, *Bayesian* theory and *Fuzzy* reasoning. On the other hand, strongly coupled algorithms are subdivided in the following groups: *recurrent* heuristic methods, *optimization-based* and *Bayesian* methods.

Next, we will briefly describe and unify the notation for each one of these methods, considering that we wish to detect an object $\mathcal{O}$ in the image, by evaluating n features described by $\mathbf{x}_1, \ldots, \mathbf{x}_n$ (representing for instance color, contour, texture, movement, etc.):

- **Weakly coupled algorithms**
  - **Weighted average**: The simplest method of feature fusion is to take a weighted average of the estimate done for each cue. If $p(\mathcal{O}|\mathbf{x}_i)$ represents the posterior object membership probability for cue $\mathbf{x}_i$, the cues are combined to obtain the overall value of object membership (belief) using a weighted sum of its respective responses:
  $$Bel(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathcal{O}) = \sum_{i=1}^{n} \omega_i p(\mathcal{O}|\mathbf{x}_i) \tag{4.1}$$
  This data fusion scheme is known as a *linear opinion pool* (48).
  - **Voting**: In the general approach for the fusion of n features $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ using the voting approach, the estimate done for each cue (for instance the posterior probability $p(\mathcal{O}|\mathbf{x}_i)$) is binarized according to a particular threshold T, so that each feature $\mathbf{x}_i$ produces a binary vote v_i at each image pixel location $\mathbf{u}$, for the single class $\mathcal{O}$ (i.e., $v_i(\mathbf{u}, \mathcal{O}) = 1 \Leftrightarrow p(\mathcal{O}|\mathbf{x}_i) > T$). Subsequently, the decision of membership to the $\mathcal{O}$ class may be done considering several schemes (101):
    - Unanimity $\sum_{i=1}^{n} v_i(\mathbf{u}, \mathcal{O}) = n$
    - Byzantine $\sum_{i=1}^{n} v_i(\mathbf{u}, \mathcal{O}) > \frac{2}{3}n$

* Majority $\sum_{i=1}^{n} v_i(\mathbf{u}, \mathcal{O}) > \frac{n}{2}$
* m-out-of-n $\sum_{i=1}^{n} v_i(\mathbf{u}, \mathcal{O}) > m$

For each of the schemes, the image pixel $\mathbf{u}$ is assigned to the class $\mathcal{O}$ if the previous conditions are satisfied.

– **Bayesian Methods**: The Bayesian approach provides an elegant way of formulating the feature integration in terms of probability theory.

From Bayes theorem (11), the posterior object probability when features are considered independent may be expressed as:

$$p(\mathcal{O}|\mathbf{x}_1, \mathbf{x}_2) = \frac{p(\mathcal{O}, \mathbf{x}_1, \mathbf{x}_2)}{p(\mathbf{x}_1, \mathbf{x}_2)} = \frac{p(\mathbf{x}_1|\mathcal{O})p(\mathbf{x}_2|\mathcal{O})p(\mathcal{O})}{p(\mathbf{x}_1, \mathbf{x}_2)} \tag{4.2}$$

This data fusion scheme is known as an *independent opinion pool* (48).

– **Fuzzy Reasoning**: Fuzzy logic (146) is also used to perform the fusion of several object features, in such a way that the degrees of belief of each feature to the object or background classes, are represented by membership functions. The membership function $\mu_{\mathbf{x}_i}$ gives a membership value $\mu_{\mathbf{x}_i}(\mathbf{u})$ for each image pixel when evaluating feature $\mathbf{x}_i$.

The fusion of the responses of all features is done using the fuzzy composition operation. One possible definition of this operator is through the '$\min$' operation:

$$\mu_{\mathbf{x}_1}(\mathbf{u}) \circ \mu_{\mathbf{x}_2}(\mathbf{u}) \circ \ldots \circ \mu_{\mathbf{x}_n}(\mathbf{u}) = \min(\mu_{\mathbf{x}_1}(\mathbf{u}), \mu_{\mathbf{x}_2}(\mathbf{u}), \ldots, \mu_{\mathbf{x}_n}(\mathbf{u})) \tag{4.3}$$

- **Strongly coupled algorithms**

– **Recurrent heuristic methods**: Recurrent heuristic techniques involve those algorithms in which the prior used by a module can be affected by the output of another module (which can itself be affected in some way by the original module), but the fusion is performed by heuristic or ad-hoc rules, and not by rigorous optimization or probabilistic methods. We include into this category those algorithms with a feedforward relation, where the interaction between different cues is performed sequentially.

– **Optimization-based methods**: Rather than relying on heuristic rules, some approaches use optimization techniques when integrating object cues. These techniques do not require to know a priori the relation between cues. That is, the whole set of parameters describing each one of the cues is combined in specialized algorithms to iteratively search the state of the features that best describe the target. For instance, let the cue $\mathbf{x}_1$ be parameterized by $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]$ and the cue $\mathbf{x}_2$ be parameterized by $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_n]$. By considering the image at time $t-1$, $\mathbf{I}^{t-1}$ as the reference image and the image at time t, $\mathbf{I}^t$ as the image where the target needs to be located, the goal of the optimization-based approaches is to minimize (over parameters $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$) an energy function like:

$$E(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \int_{\mathcal{O}^{t-1}} \|\mathbf{I}^t(\boldsymbol{\lambda}, \boldsymbol{\nu}) - \mathbf{I}^{t-1}\| \tag{4.4}$$

where $\mathcal{O}^{t-1}$ refers to the object pixels in image $\mathbf{I}^{t-1}$.

– **Bayesian Methods**: Our final class of strongly coupled techniques involves those algorithms in which the relation between cues is formulated through the Bayes' theorem. Whereas in the Bayesian weakly coupled methods, the features where considered mutually independent, now they are assumed to be conditionally dependent. This is expressed by:

$$p(\mathcal{O}|\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_2|\mathcal{O}, \mathbf{x}_1)p(\mathbf{x}_1|\mathcal{O})\frac{p(\mathcal{O})}{p(\mathbf{x}_1, \mathbf{x}_2)} \tag{4.5}$$

$$= p(\mathbf{x}_1|\mathcal{O}, \mathbf{x}_2)p(\mathbf{x}_2|\mathcal{O})\frac{p(\mathcal{O})}{p(\mathbf{x}_1, \mathbf{x}_2)} \tag{4.6}$$

Equations 4.5 and 4.6 differentiate the cases where feature $\mathbf{x}_2$ depends on feature $\mathbf{x}_1$ (Eq. 4.5), or that $\mathbf{x}_1$ depends on $\mathbf{x}_2$ (Eq. 4.6).

4.3 State of the art in fusion of visual modules

In this section we will review the state of the art of the tracking and figure ground segmentation techniques that make use of multiple object cues. Note that figure ground segmentation in video sequences is closely related to the tracking problem. The difference between both techniques refers to the fact that whereas in tracking the goal consists on recovering the position of the object points throughout time, in figure-ground segmentation the goal is to provide a dense map of object membership. Therefore, figure-ground segmentation can be interpreted as an

Weakly coupled fusion algorithms			
Weighting average	**Voting**	**Bayesian**	**Fuzzy**
Prokopowicz'94 (105)	Nordlund'97 (99)	Toyama'00 (130)	Jadon'01 (53)
Crowley'97 (25)	Bräutigan'98 (19)	Sherrah'01 (114)	Kragić'01 (64)
Birchfield'98 (14)	Pirjanian'98 (104)	Hayman'02 (48)	
Khan'01 (61)	Kragić'01 (64)	Taylor'03 (124)	
Shearer'01 (112)		Leichter'04 (67)	
Triesch'01 (131)			
Hayman'02 (48)			

Table 4.1: Subclasses into the weakly coupled category.

initial stage before solving the tracking. Consequently, in this review we will include both tracking and figure-ground segmentation references.

Nevertheless, since we are only concerned on methods dealing with video sequences, we will discard from our analysis all those references for static image segmentation. The reason is that the techniques used in single image segmentation do not need to impose the temporal consistency constrains commented above, and they can yield very different results when applied to two consecutive (and similar) frames of a video sequence.

Next, the reviewed papers will be briefly described and classified according to the proposed taxonomy. Tables 4.1 and 4.2 summarize the works that will be referenced in the following sections.

4.3.1 Weakly coupled algorithms

4.3.1.1 Weighted average

Weighted average methods for cue integration is one of the simplest approaches. It is only applicable when all the cues provide redundant information, that is, all of the cues provide the position (or membership probability) of the object in the image. Subsequently, the fusion method takes a weighted average of this redundant information.

The simplicity of the methodology allows real-time applications. Birchfield (14), integrates color histograms and edges in order to track a person's head. The integration is performed through a weighted average approach, where the weights assigned to the estimation of each feature are equal and kept constant.

More robust approaches are those that adapt the weighting parameters, depending on the reliability of each feature. There are some methods that use binary weights, and only one of

Strongly Coupled fusion algorithms		
Recursive	**Optimization-Based**	**Bayesian**
Azoz'98 (4)	Wren'97 (139)	Isard'98 (51)
Darrel'98 (26)	Cootes'98 (24)	Rasmussen'98 (107)
Isard'98 (52)	Hager'98 (45)	MacCormick'00 (76; 77)
Beymer'99 (12)	Shi'98 (116)	Khan'04 (60)
Toyama'99 (129)	Vetter'98 (132)	Wu'04 (142)
Kruppa'01 (65)	Shi'00 (117)	Moreno-Noguer'05 (85)
Sherrah'01 (115)	Tao'00 (125)	
Moreno-Noguer'02 (92)	Matthews'04 (80)	
Siebel'02 (118)		
Nummiaro'03 (100)		
Shen'03 (113)		
Spengler'03 (122)		
Moreno-Noguer'04 (88)		

Table 4.2: Subclasses into the strongly coupled category.

the features is used for each frame. For instance, Prokopowicz *et al.* (105) propose using one out of several cues, color, motion, disparity and a template matching, for tracking tasks. The selection of the appropriate cue for a specific target tracking is done based on heuristic and empirical reasonings.

In the same way, the work presented by Crowley and Berard (25), cycles between three different algorithms when tracking a face. The different algorithms are based on color histograms, template matching and blinking detection, respectively. When a global confidence factor decreases, the color module is used, because it is more robust (although less accurate than template matching). Otherwise, if the global confidence factor is high, the tracking is performed through the template matching, which is more accurate but less robust. The blinking module is used to detect the eyes and re-initialize the template and color histogram. A similar idea is used by Shearer *et al.* (112). Two trackers are run in parallel, a region tracker based on color correlation and a contour tracker. The success of each algorithm is computed by separate considering the consistency of the tracking results with previous frames. When both trackers succeed, their response is combined, and when one of the trackers fails, its state is updated only considering the response of the successful tracker.

Other approaches use adaptative and real valued weights. Khan and Shah (61) propose a Maximum a Posteriori (MAP) framework for video segmentation using color, motion and a

position prediction cue. This is a general approach that permits segmenting the image in more than 2 classes (not only foreground and background). For instance, in the experiments shown in (61), each sequence frame is segmented into 6 classes. Although the method has a Bayesian background, we consider it to be a 'weighted average technique', as it assigns heuristic weights to each feature, which are adjusted at each frame, based on some confidence measures.

The *democratic integration* scheme, introduced by Triesch and Malsburg (131) adaptatively integrates different cues in a self-organized manner. The architecture is tested in a face tracking experiment. Motion, color, position prediction, shape and contrast make independent assumptions about the target position, and a global estimation is obtained by a weighted average of these assumptions. Subsequently each cue adapts toward the result agreed on, adapting both the parameters describing the state of the cues, as the weights of the integration scheme. Although the cue models used are quite simple, the complete system is able to overcome several difficulties such as occlusions, abrupt changes of illumination and non-linear target dynamics. However, the system is not tested in highly cluttered environments.

Hayman and Eklundh (48) present a similar approach. Although in their paper, the methodology is considered as a voting approach, the cue combination strategy follows the formulation of a weighting scheme. In this work, motion, color, contrast and prediction cues are integrated in order to perform figure ground segmentation of video sequences. Temporal consistency is guaranteed by learning at each time step the models for color and contrast cues based on the output of the overall algorithm in previous frames.

4.3.1.2 Voting

Voting algorithms are commonly used for integrating 'simple' visual cues when real-time tasks are required, and when the model for each one of the cues is not available (therefore it is not possible to validate the state of the cue by comparing with a model). Pirjanian *et al.* (104) perform the real time tracking and control of a stereo head mounted on a pan tilt unit, by integrating a blob tracking, image differencing, edge tracking and template matching modules. The fusion is performed through a voting approach using a *majority* scheme. A similar approach is presented by Kragić and Christensen (64) for visual servoing and robot end-effector tracking tasks. In this approach an *m-out-of-n* voting scheme is used to integrate edges, disparity, color, correlation with previous image and motion estimation. Bräutigan *et al.* (19) also use an *m-out-of-n* voting scheme in order to integrate the response of several vision modules, for segmenting planar regions of images. The visual modules used in this work are point based disparities,

perspective distortion of texture patterns, algebraic invariants of five matched points, grey level homogeneity, and monocular junction angles.

Nordlund and Eklundh (99) use two independent dense feature domains (motion and depth from stereo) to produce a 2D histogram, i.e, the coordinate axis of the histogram are the pixel motion and depth, respectively. Peaks in the histogram represent areas in the image which have approximately constant values in both feature domains at the same time. Backprojecting these peaks into the image, different image regions can be segmented. Note that this fusing scheme is exactly the same than a voting approach in the *unanimity* modality, that is, each feature produces a binary vote at each pixel location for a single class (or image region), and a pixel is considered to belong to a specific region R_i if all of the features have voted for region R_i.

4.3.1.3 Bayesian theory

Bayesian theory provides a formalism to the fusion problem that allows the information of several sources to be combined according to the rules of probability theory with the well-known Bayes rule. One work that presents this formalism in the figure-ground segmentation field is (48), by Hayman and Eklundh. In this work, the Bayesian approach to segmentation is described (together with a weighted average approach -commented above-), and used to integrate motion, color, image contrast and position prediction cues. For the experiments considered in the paper, both techniques (Bayesian theory and weighted average) perform well. The reliabilities of each cue are incorporated through hyper-priors on the model parameters. An additional aspect of this approach is in suggesting mechanisms for suppressing cues when they are unreliable.

Bayesian networks (55), permit to represent efficiently probabilistic dependencies between the true state of the target and the measures obtained from different cues. Toyama and Horvitz (130), address the problem of multiple visual cue integration through a process of Bayesian Modality Fusion (BMF). BMF uses a Bayesian network to combine the estimates of several complementary and independent object features: color, motion, and background subtraction. A reliability indicator for each feature is incorporated into the network, similarly to the 'democratic integration' approach (131). The difference is that using Bayesian networks, the whole approach can be formulated in a probabilistic way. Sherrah and Gong (114) extend the BMF into the so called Continuous Global Evidence-Based BMF (CBMF). The main difference between the BMF and CBMF is that the latter uses continuous rather than discrete variables, relieving the complexity of inference. CBMF is used in (114) for tracking a human head by integrating skin

color, motion (through frame differencing) and shape (through an ellipse fitting procedure). The reliabilities of each feature are also introduced in the network in order to be adapted.

An example of Kalman filtering applied to fusion of visual cues is presented by Taylor and Kleeman (124), which integrate color, edges and texture cues in order to estimate the pose of an object. The set of cues conform the measurement vector of the filter and the pose of the object is parameterized in the state vector to be predicted and estimated using the Kalman framework.

On the other hand, particle filters (see Section 2.3) realize the Bayes filter updates according to a sampling procedure (often referred to as sequential importance sampling). This makes them useful for representing arbitrary probability densities, outperforming the performance of the Kalman filter which are rectricted to Gaussian distributions and linear dynamic models. In that sense, Leichter *et al.* (67) propose a probabilistic framework to integrate multiple independent tracking algorithms. The method is general in that it may be used for combining any set of tracking algorithms that provide a Probability Density Function representation of the state of the target, and in particular, it allows to combine multiple particle filters. The effectiveness of the method is demonstrated in an experiment consisting of tracking the head of a person on an indoor environment. Two particle filter algorithms are integrated, one based on color edges that tracks the contour of the head, and the other based on difference between color regions which tracks the vertical axis of the head.

4.3.1.4 Fuzzy theory

Fuzzy theory allows the uncertainty of each cue of belonging to a specific class to be represented by a continuous membership function, which expresses the degree of belief with a real number from 0.0 to 1.0. Subsequently, the fusion of all cue beliefs is realized through the fuzzy inference rules. For instance Kragić and Christensen (64) use a fuzzy approach for tracking the end-effector of a robot arm, in a visual servoing application. Edges, disparity, color and correlation are the cues utilized. In this work, fuzzy methods are compared with the voting algorithms, and for the same experimental conditions it is observed that voting algorithms perform better than methods based on fuzzy theory.

In (53) Jadon *et al.* develop a fuzzy-logic-based framework for detecting abrupt changes and gradual changes between consecutive frames of a video sequence. The used features are a measure (between two consecutive frames) of histogram intersection, pixel difference and difference in the number of edge-pixels. Finally, these features are used to design fuzzy rules and decide if a frame transition is affected by an abrupt or gradual change.

4.3.2 Strongly coupled algorithms

4.3.2.1 Recurrent heuristic methods

The major part of the fusion techniques in the tracking and figure-ground literature are based in a recurrent heuristic scheme, where some cues generate constrains for other cues. Most of these techniques are 'application specific', and use *ad-hoc* rules when performing the cue integration. For example, Azoz *et al.* (4) use color, motion and shape to track the human arm dynamics. The integration method is based on a sequence of heuristics, which start with a rough segmentation of the arm regions using the color module. Motion and shape are subsequently applied to reduce the number of false blobs, and finally, a set of connectivity constrains between the remaining blobs allow to find the shoulder, elbow and hand.

In the approach of Beymer and Konolige (12), depth from stereo and an intensity template is used to track multiple persons. Initially, stereovision is applied in order to discriminate the foreground objects depending on its depth, and subsequently a correlation with a grayscale template allows the tracking of a specific target.

Darrel *et al.* (26) integrate several features for robust face tracking in the event of develop a virtual mirror interface which can react to people and combine the user's face with several graphical effects. Stereo, color and grey-scale pattern matching modules are integrated into a real-time tracking system. Stereo processing allows to isolate the figure of the user from other objects in the background. Its result feeds into a skin-hue classifier which identifies and tracks body parts. Finally, stereo and color results are used by the pattern matching to localize the face within a region containing the head.

In one of the works derived from this dissertation (92) we integrate color histograms and depth from stereo using a sequential scheme, for tracking faces in real-time (the system runs at more than 30Hz). A first search of an elliptic shape with skin-like color is performed by color histogram intersection. The result feeds into a stereo module, which computes the depth of the face candidate and updates the size of the ellipse for the next iteration.

Another method specifically tailored to the particular application of people tracking is described by Siebel and Maybank (118). This work executes in parallel multiple algorithms, a motion tracker, a region tracker, a head detector and a contour tracker. Each one of the algorithms is designed for a specific sub-task and is dependent on the output of other modules for initialization or correction.

Kruppa and Schiele (65) combine several features (color, template and contour) for the purpose of detecting a face in an image. The particularity of the method is that the integration of the features is performed in a sequential hierarchical way: first color and the template are used in parallel in order to find two separate probability maps of the face. Next, these maps are combined with the contour information to obtain a set '1' of face candidates based on the color/contour information, and a set '2' of face candidates based on the template/contour information. Finally, the faces are detected considering the face candidate that maximizes the mutual information of the previous sets '1' and '2'.

These previous methods are so 'custom-made' for a specific application that do not allow to integrate extra cues. A more general approach is the *Incremental Focus of Attention* (IFA) framework proposed by Toyama and Hager (129), which combines several tracking algorithms into a layered hierarchy. The idea behind IFA has close resemblance to the idea presented by Crowley and Berard (25), i.e, when conditions are good tracking is performed through accurate although less robust algorithms, and as conditions deteriorate, tracking is performed through more robust, yet less accurate algorithms. Besides proposing the fusion scheme, (129) presents an experiment of tracking a face, where from low to high accuracy, the object cues that are used are the color, motion, and shape (templates of several face features).

There are a set of works that do not exploit completely the strength of the particle filters, as the multi-hypotheses framework is only used to estimate the change of a unique cue (for instance the target position) and these hypotheses are weighted considering the state of other features (for instance color and shape), which are approximated using only one hypothesis. That is, just the position cue is actually estimated by the particle filter formulation. Although particle filters are inspired on a Bayesian framework, this kind of approaches have a sequential nature, thus belonging to the recurrent methods in the proposed taxonomy of fusion methods. Along these lines, Nummiaro *et al.* (100) propose an adaptative color-based particle filter for tracking. The target is modeled using an ellipse, and its position is estimated by multiple hypotheses, which are weighted according to the color cue, by comparing their histogram with a model histogram using the Bhattacharya distance.

In the same manner, Spengler and Schiele (122) extend the *democratic integration* (131) scheme to a multi-hypothesis framework by weighting the different target position hypotheses evaluating separately color and shape, and keeping the contribution of both cues constant. A more robust approach is presented by Shen *et al.* (113), in which the contribution of each cue is adapted depending on cue reliability.

Sherrah and Gong (115) use the Condensation algorithm (51) to track the pose (position and orientation) of a head. Each one of the position and orientation hypotheses is weighted according to a reliability measure which takes into account the similarity with respect to a set of templates (images acquired off-line for different individuals and different head poses) and also, with respect to skin color information.

The ICondensation approach, by Isard and Blake (52), uses a slightly different approach to integrate the color cue into the Condensation algorithm when tracking hands. Color information is used as a prior to bias the shape hypotheses generation stage, and the individual samples are weighted only based on edge data.

In order to enhance the robustness of the figure-ground segmentation with respect to illumination changes, in our previous work (88) we use a particle filter to estimate the color distribution of the target, which is parameterized by a Mixture of Gaussians. The multiple hypotheses are only generated about the color cue, and the corresponding weights are assigned taking into account the shape information. The color estimate allows to segment the target from the background, and subsequently the shape of the target is updated using deformable contours. The snake fitting procedure is highly simplified after the figure ground segmentation is performed using the color estimate.

4.3.2.2 Optimization-based algorithms

Recurrent methods use ad-hoc methodologies for fusing the different cues, many times relying on heuristics and even manual tuning of parameters. On the other hand, optimization-based approaches offer a general framework, where cue integration is performed through the minimization of functions of energy including the state of all the cues. These methods are less 'application-dependent', and do not require to know the relation between features. For instance, Hager and Belhumeur (45) propose a region tracking algorithm able to cope with simultaneous changes of shape and appearance. Geometric distortions of the target are accommodated by introducing a motion model to the target, parameterized by affine deformations, and changes in appearance are accommodated by incorporating illumination models, represented by linear combinations of a small number of 'basis' images (obtained by applying Singular Value Decomposition (SVD) to a large training set of images). The parameters of the affine deformations and the weights of each basis image (subspace coefficients) are simultaneously optimized by a least squares procedure.

A similar approach to this work are the Active Appearance Models (AAMs), by Cootes *et al.* (24) and Matthews and Baker (80). AAMs model shape (defined by a mesh) and appearance (defined by a set of basis images, as in (45)) separately, and are able to deal with simultaneous geometric and appearance changes of the target. The adaptation process is achieved through the minimization of an energy function involving the parameters of the mesh and the subspace coefficients, usually by gradient descent algorithms. In (132), Vetter and Blanz extend the AAMs and integrate the target 3D shape and appearance. The only difference with previous methods is that in (132), the shape is modeled through a 3D mesh.

Wren *et al.* (139), present the 'Pfinder', an application in which color and shape are integrated in order to track a person's body. The person and background are modeled by its color and shape (pixel positions inside the blob representing the person or the background) using Gaussian models. Subsequently, given a new image, each pixel is classified to the object and background classes, according to its distance with respect to the models. Furthermore, at each iteration the person and background models are updated.

Tao *et al.* (125) propose a dynamic layer representation for tracking moving objects (in particular they perform a vehicle tracking and segmentation from airborne cameras). Each layer is approximated through three components, namely, motion, shape information and appearance, which are estimated simultaneously in a maximum a posteriori framework. For each image sequence, the optimal solution is iteratively computed using the generalized Expectation Maximization (EM) algorithm.

Shi and Malik (117), introduce the *normalized cuts*, a global technique for single image and video segmentation. The segmentation process is performed through a graph partitioning technique. Each image pixel represents a node of a graph and each pair of pixels are connected by a graph edge. The weight of the edges connecting two pixels, reflects the likelihood that the two pixels belong to one object, and can be computed by integrating any kind of image information, such as brightness, color, texture and motion. In order to perform the segmentation, the *normalized cuts* technique partitions the graph in disjoint parts such that the dissimilarity between the different parts is maximized. When using the method for segmenting video sequences (116), temporal consistency is guaranteed by connecting the nodes that are in a spatiotemporal neighborhood. However, the drawback of this and related algorithms is that they are slow.

4.3.2.3 Bayesian theory

Bayesian theory, also allows the integration of the information provided by several cues, when there is some kind of probabilistic dependence between the cues. Thus, for example, Rasmussen and Hager (107) propose a tracking method that combines color and edge information, using a Joint Probability Data Association Filter (JPDAF). JPDAF improves the performance of the PDAF filters (8) (extension of the Kalman Filter when multiple measurements of the same event are validated) in that it incorporates an exclusion principle, which allows to share information among separate PDAF trackers and prevents them from latching onto the same target.

As we have previously stated, particle filters, have been demonstrated to be robust enough for tracking objects that move with complex dynamics. Therefore, in order to achieve robust tracking, some approaches have integrated appearance and shape in a particle filter framework, where object appearance is represented by a set of basis images (usually collected using PCA), which are linearly combined through the subspace coefficients. Then, the simplest approach to incorporate the appearance into the particle filter, is to augment the state space of the target, with the subspace coefficients (Isard and Blake (51)). Note that now, both features are estimated using the particle filter framework, and not only one of the features, as was done in the particle filter recurrent approach of the previous section. However, as it was observed by Khan *et al.* in (60), to proceed by simply augmenting the state space is problematic because it suffers from the curse of dimensionality problem (the number of samples in the particle filter increases exponentially with the dimensionality of the state vector). The approach in (107), just presented above, also suffers from the same problem. (60) proposes to use a Rao-Blackwellized particle filter, where the subspace coefficients are integrated out of the state vectors. This procedure reduces considerably the number of samples.

Partitioned sampling, introduced by MacCormick (75; 76; 77) is another technique which allows to reduce the curse of dimensionality problem when using particle filters. The basic operation of the method consists of applying the 'hypotheses generation' and 'hypotheses correction' stages (see Section 2.3 for a description of these operations), independently for different parts of a high dimensional state vector. This reduces considerably the region of the state vector space where samples are propagated, and as a consequence, reduces also the number of needed samples. A partitioned sampling technique can be easily adapted in order to create a framework for integrating several object features in a tracking or figure-ground segmentation

task. Nevertheless, in the original works, this technique has not been used for integrating several features, but for tracking multiple objects (76) and articulated objects (77), in both cases just considering the edges as a unique object feature.

Closely related to partitioned sampling is the approach proposed by Wu and Huang (142), where two object features, color and shape (represented by an ellipse) are integrated in order to track human faces in indoor environments. Although both features are used in a unique particle filter, the prediction and observation stages are done separately and sequentially, in order to reduce the curse of dimensionality problem previously commented.

In the next chapter of our dissertation, (and the associated work presented in (85; 87)) we extend this formulation and propose a framework to integrate any number of (conditionally dependent or conditionally independent) algorithms, whose output approximates the state of the tracked object, and it is represented by a Probability Density Function (PDF). For instance, it permits to combine an algorithm using Kalman filtering that represents the target state through a Gaussian PDF, with another algorithm using a particle filter that represents the target state through a general PDF. In particular, in the work described in (85), we combine three of the object features presented in Chapter 3, namely the color, the contour and the Fisher colorspace. All of the features are estimated through a particle filter formulation and integrated with the proposed framework, allowing for a robust figure-background segmentation of rigid an non-rigid objects in highly cluttered environments, with abrupt changes of both the target's position and appearance. This approach also extends the work of Leichter *et al.* (67), only valid for combining conditionally independent features.

4.4 Analysis

It is difficult to compare the different algorithms described in the previous sections because they are based on different assumptions and tested under different experiments. However, we can give general comments and emphasize the main advantages and disadvantages for each one of the groups conforming the proposed taxonomy. This analysis will be done in terms of some of the 'desired properties' of the fusion scheme, indicated in the Introduction of this chapter.

4.4.1 Robustness

We consider that the robustness of a system is related to the ability of dealing with challenging and uncontrolled environment conditions, such as abrupt changes of illumination, cluttered

backgrounds or non-linear dynamics of the target. As a global view, strongly coupled algorithms perform better than weakly coupled, and exploit better the capabilities of each feature, since usually, the performance of a specific module is enhanced when using the information of another module as a prior. For instance, although the weighting average approach of the *democratic integration*, by Triesh and Marlsburg (131) (weakly coupled) succeeds in tracking when there are abrupt illumination changes, the experiments shown by Khan *et al.* (60), (strongly coupled, with a Rao-Blackwellized particle filter) are much more compelling.

Within the weakly coupled approaches, the fuzzy based fusion is probably the less robust approach. In particular, Kragić and Christensen (64) compare voting and fuzzy approaches, in which the former shows a considerably smaller tracking errors. The capabilities of averaging, voting and Bayesian (when using single hypotheses) approaches is quite similar, as noted by Hayman and Eklundh (48). We would like just to make a special mention with respect to the Bayesian scheme for integrating multiple independent particle filters proposed by Leichter *et al.* (67). Even though the examples shown in their work are not particularly challenging, we believe that the considered framework is promising in terms of robustness. The most outstanding attributes of this approach are that it is general and mathematically justified, and may be used for combining any set of tracking algorithms (based on different visual modalities) that provide a PDF estimate of the target. However, (67) assumes independence of the individual algorithms which are integrated, and does not allow to exploit a possible dependence between the combined visual modalities.

With respect to the strongly coupled algorithms, the recurrent approaches are methods built for specific applications, using *ad-hoc* rules and hand-tuning of parameters. This makes them difficult to be extrapolated to new operation conditions, and therefore, less robust. Moreover, since cues use to be sequentially related, these are methods prone to be sensitive to anomalous or uncertain cues. Optimization based integration algorithms, allow to design more robust approaches; for instance Hager and Belhumeur (45), Cootes *et al.* (24) and Matthews and Baker (80), propose tracking methods able to cope with changes of both geometry and illumination. The problem of these approaches is that they require of models that fit the data and prior distributions of possible results (for instance a set of basis images spanning all possible results). Similar inconveniences are suffered by Bayesian approaches based on a single hypothesis. This might be overcome by Bayesian approaches with a particle filter formulation, since the fact of using multiple hypothesis relaxes the need of a very accurate model. For instance, Khan *et*

al. (60) (with a Rao-Blackwellized particle filter) impressive results in challenging environments. In next chapter, we will show that the probabilistic integration framework proposed in this dissertation, offers a general methodology, which may be particularized to integrate object features estimated by conditional dependent particle filters, permitting the tracking in highly cluttered and dynamic environments.

4.4.2 Adaptability

The adaptability of the fusion scheme with respect to changes in the target and background is a key issue for achieving robustness. Fusion methods typically use two methodologies for adaptation: adjust the reliability associated with each cue or adapt the model describing a particular feature of the target (2) (or a combination of both methods).

Weakly coupled schemes adapt the cues by adjusting their reliabilities in the fusion scheme. For instance, Crowley and Berard (25), Prokopowicz *et al.* (105) and Shen *et al.* (112), use one over several tracking modalities (each modality associated to a different object cue), depending on the reliabilities of the cue for a specific experimental conditions. Triesh and Malsburg (131), adjust the weights associated to each feature depending on some measure of quality. Furthermore, this approach updates the models describing the cues, using a linear rule. Toyama and Horvitz (130) and Sherrah and Gong (114) introduce into the Bayesian networks a reliability indicator for each feature, which needs to be inferred. In (48), Hayman and Eklundh integrate the reliabilities into a Bayesian fusion scheme through hyper-priors on the model parameters.

While weakly coupled fusion approaches are mostly based on an adjustment of the reliabilities associated to each cue, strongly coupled methods update the model describing the features. For instance, Nummiaro *et al.* (100), and Shen *et al.* (113) adapt the color feature by adjusting a color histogram to match histograms of the target using a leaking integrator. More robust approaches are those where the model parameters describing the change of the cues are incorporated in the estimation process. For instance, in the optimization-based techniques proposed by Hager and Belhumeur (45), the parameters describing an affine deformation and an appearance change of the target, are inferred through the estimation process, and therefore are automatically updated. Even more robust techniques are proposed by Khan *et al.* (60), Wu and Huang (142) and the method proposed in this thesis (and the corresponding work in (85)), where appearance and geometric parameters describing the target state are integrated and updated through a particle filter framework. With a multiple hypotheses framework, different

configurations of the model parameters are considered at each time step, providing robustness to unexpected changes on the target state.

4.4.3 Modularity, complexity and scalability

Modularity in the design of the fusion scheme is an interesting capability, since it permits to reduce the complexity of the overall integration process. Furthermore, modular systems are easily scalable with extra features.

From this point of view, weakly coupled fusion approaches have a much more modular structure than strongly coupled methods. This allows to use weakly coupled schemes for implementing real time applications. For instance, Birchfield (14) tracks human faces using an 'average weighting' fusion approach. The applications proposed by Prokopowicz *et al.* (105), Crowley and Berard (25), and Triesh and Malsburg (131) run also in real time. Voting and fuzzy approaches are even more efficient, since they are model-free and do not require a precise mathematical model of the controlled process. Kragić and Christensen (64) implement a real time visual servoing system using both schemes.

Although strongly coupled methods perform more robustly than weakly coupled ones, their structure is complex and the flow of information does not follow linear paths, what makes difficult the tasks of replacing or introducing modules. This is specially reflected with the recursive and optimization based approaches. As an example, Siebel and Maybank (118) combine a set of visual modules using intricate relations of dependence, valid only for a specific application. The recursive methods proposed by Azoz *et al.* (4), and Beymer and Konolige (12) are designed in the same way. Even though optimization-based approaches have a less intricate structure, they rarely work in real time, since these techniques are based on iterative procedures (for instance gradient descend, in (24; 45; 80), or a graph partition technique in the *normalized cuts* (117), by Shi and Malik). Similar complexity is observed when integrating several features in a conventional particle filter framework, as is done by Isard and Blake (51). However, the Rao-Blackwellized particle filter (Khan *et al.* (60)), and the partitioned sampling (MacCormick and Blake (76; 77)) relax the high computational cost of using particle filters with high dimensional state vectors. Inspired in the partitioned sampling, the work proposed in this dissertation and described in next chapter (and in (85)) we propose a framework to integrate any number of (conditionally dependent or independent) algorithms, where the complexity of the system does not explode when introducing extra visual modules.

4.5 Summary

Tracking and figure-ground segmentation in video sequences is a topic of significant interest in a wide variety of computer vision tasks, extending from video coding to mobile robot navigation.

It is clear that the fusion of the information from multiple object features improves the performance of such algorithms. For instance, one of the conclusions of the extensive and popular survey on tracking algorithms presented by Moeslund and Granum (82), states that:

"*For future systems to be more successful and less dependent of various assumptions new methods and a combination of current methods should be developed, i.e., the combination of various image cues, such as motion and silhouettes, and more extensive and adaptive use of human models. Furthermore, new sensors or combinations of sensors might also be an interesting path into the future.*"

In this chapter we have reviewed the recent development on the research of integration of visual object features for figure ground segmentation. We have presented a taxonomy, inspired in the sensor fusion classification proposed by Clark and Yuille (23). Two major integration categories are distinguished, namely weakly coupled and strongly coupled, depending on the degree of interaction between the vision modules. In the former group, we include weighting average, voting, Bayesian with independent cues and fuzzy integration approaches. Strongly coupled schemes, are classified into recurrent, optimization-based and Bayesian with correlated cues techniques. Almost 50 papers are reviewed and classified into the proposed taxonomy.

We have observed that most of the experiments in the reviewed papers, deal with highly constrained environments. Challenging experimental conditions, like cluttered background, abrupt changes of illumination, occlusions, template deformations or non-linear dynamics, are only addressed by a reduced number or works. Furthermore, most of the works are designed for specific applications, usually for computer-human interaction tasks and tracking parts of the human body (hands, face, whole body, arms...).

Consequently, one of the main goals in future works should be to build more general fusion approaches, not restricted to specific problems, and improve their robustness in non-stationary environments, affected by the disturbances previously mentioned. Real time is another factor that must be taken into account.

Among all the reviewed techniques, the most promising are the fusion approaches based on a strongly coupled bayesian technique. Specifically, the methods fusing dependent particle

filters are those that have showed better performances. For instance, the Rao-Blackwellized particle filter framework proposed by Khan *et al.* (60), presents the challenging problem of tracking an unmarked honey bee in an observation hive. Temporary occlusions, complex variations in the appearance and unpredictability of the bee's movements are some of the difficulties that need to be addressed by integrating appearance and geometry.

In next chapter we will present a framework that permits to deal with most of the limitations of the reviewed papers. The methodology that we propose, provides a probabilistic framework to integrate as many features as necessary, allowing the features to be both conditionally dependent or independent, and described by *PDF*'s. In particular, this framework permits to combine several particle filters, with other algorithms which output is a *PDF*, like Kalman filters. Furthermore, since each object feature is attached to a different state vector, the computational cost of the whole system is significantly lower than including all the features in a unique state vector. Tracking results in highly cluttered environments, with abrupt changes of illumination and object position will demonstrate the effectiveness of the method.

Chapter 5

Probabilistic framework for integrating multiple cues

Remind the main goal of the dissertation: propose a framework to integrate multiple object cues, allowing to estimate the configuration of the target through video sequences which might suffer from different artifacts, such as, abrupt changes of illumination, cluttered backgrounds and non-linear dynamics. In previous chapters we have established the bases that will be used to describe the proposed framework: Chapter 2 introduced the probabilistic algorithms (Kalman and particle filters) used to estimate the state of the individual features. Chapter 3, defined the features that will be used to represent the target, and Chapter 4 reviewed and analyzed previous approaches in the computer vision literature that perform tracking based on multiple object features.

The present chapter incorporates all this information in order to propose a new technique for fusing multiple cues to robustly segment an object from the background (and subsequently track it), in video sequences that suffer from the artifacts just mentioned. Robustness is achieved by the integration and interaction of the appearance and geometric object features described in Chapter 3, and by their estimation using non-linear particle filters (described in Chapter 2). Most of previous approaches reviewed in Chapter 4, assume independence of the object cues or simply apply a particle filter formulation to only one of the features, and assume a smooth change in the rest, which can prove very limiting, especially when the state of some features needs to be updated using other cues or when their dynamics follow non-linear and unpredictable paths.

The technique presented here offers a general framework to integrate as many features as necessary, the state of which is approximated via a Probability Density Function (PDF). Apart

from permitting to combine several particle filters, the method is also valid for integration of features estimated using the Kalman filter, and in general any algorithm which outputs a PDF and satisfies a 'hypotheses generation - hypotheses correction' scheme (as described in Chapter 2). Besides being analytically justified, the proposed approach is applied to develop a robust tracking system that adapts online and simultaneously the colorspace where the image points are represented, the color distributions of the object and background, and the contour of the object. Results with synthetic data and real video sequences demonstrate the robustness and versatility of our method.

5.1 Introduction

The integration of several visual features has been commonly used to improve the performance of the techniques for tracking and figure-ground segmentation in video sequences. In the previous chapter we have reviewed the most relevant and recent works in this field. Based on the classification proposed by Clark and Yuille in (23), the fusion methodologies have been classified into the 'weakly coupled' and 'strongly coupled' techniques, where the former refers to those fusion techniques assuming cue independence, and the latter consider dependence between cues.

It has been observed that although weakly coupled methods permit to implement less costly and real time applications, the most robust results are obtained when using strongly coupled methods. In particular, the strongly coupled Bayesian methods, provide the best results, in addition to an elegant formulation of the fusion scheme.

In this chapter, we introduce a probabilistic framework that integrates several object features, which allows us to robustly segment the object from the rest of the image, in dynamically changing sequences such as the one shown in Fig. 5.1, where the central leaf is the selected target to track. Observe in the sample frames of the video sequence (top row of Fig. 5.1), some of the artifacts that might convert the tracking in a challenging task: the abrupt change of illumination between the first and second frames (which are consecutive frames of the sequence), a highly cluttered background, and unpredictable dynamics of the target movement. In spite of this, using the method that we are going to describe in the following sections, we are able to segment and track the object. The bottom row of Fig. 5.1 shows the corresponding maps of the foreground membership, where brighter points correspond to the pixels that more likely belong to the foreground.

Figure 5.1: **Video sequence affected by different artifacts that make the tracking task difficult.** Top row: Original sequence, where the central leaf is the selected target. Note the abrupt illumination change between the first and second frames (which are consecutive frames). Also, the clutter and non-linear dynamics of the leaf complicate the tracking. Bottom row: Map of foreground membership obtained using the method proposed in this chapter. Brighter points correspond to pixels classified as foreground with high certainty.

In order to perform tracking under these kind of complex dynamics, we propose a method where each one of the object features is estimated by a different algorithm satisfying the 'hypotheses generation - hypotheses correction' scheme (described in Chapter 2) and which output is represented by a PDF. As will be shown in the following sections, we will restrict to Bayesian filters such as the Kalman filter and particle filter (also described in Chapter 2).

A key consideration that must be taken into account is that we enable a conditional dependence between cues. A similar approach is presented by Leichter *et al.* (67), where several Bayesian filter algorithms are integrated for tracking tasks. However, in (67) it is assumed that the methods are conditionally independent, i.e, each algorithm estimates the state of a target feature based on some measurements which are conditionally independent of the measurements used by the other algorithms. That is, if Bayesian filter $\mathcal{BF}_1$ is based on measurements (observations) $\mathbf{z}_1$ to estimate the state vector $\mathbf{x}_1$ (representing one object feature) and Bayesian filter $\mathcal{BF}_2$ uses measurements $\mathbf{z}_2$ to estimate $\mathbf{x}_2$ (representing another object feature), for each complete state vector of the object $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2\}$ it is assumed that:

$$p(\mathbf{z}_1, \mathbf{z}_2|\mathbf{X}) = p(\mathbf{z}_1|\mathbf{x}_1)p(\mathbf{z}_2|\mathbf{x}_2) \tag{5.1}$$

Nevertheless, this assumption is very restrictive and many times is not satisfied, since it assumes that the measurements used to estimate the feature $\mathbf{x}_1$ are independent from the measurements used to estimate the feature $\mathbf{x}_2$. For instance, a usual method to weigh the samples of a contour particle filter, is based on the ratio of the number of pixels inside the contour having object color versus the number of pixels outside the contour having background color. This means that the contour feature is not independent of the color feature. In this situation if $\mathbf{z}_1$ represents the observations for the color feature and $\mathbf{z}_2$ the corresponding for the contour, the latter will be a function of both $\mathbf{x}_1$ and $\mathbf{z}_1$, i.e., $\mathbf{z}_2 = \mathbf{z}_2(\mathbf{x}_1, \mathbf{z}_1)$. Based on the definition of the conditional probability [1], it is straightforward to rewrite previous equation as:

$$\begin{aligned} p(\mathbf{z}_1, \mathbf{z}_2|\mathbf{X}) &= \frac{p(\mathbf{z}_1, \mathbf{z}_2, \mathbf{x}_1, \mathbf{x}_2)}{p(\mathbf{x}_1, \mathbf{x}_2)} = \frac{p(\mathbf{z}_2|\mathbf{z}_1, \mathbf{x}_1, \mathbf{x}_2)p(\mathbf{z}_1, \mathbf{x}_1, \mathbf{x}_2)}{p(\mathbf{x}_1, \mathbf{x}_2)} \\ &= p(\mathbf{z}_2|\mathbf{z}_1, \mathbf{x}_1, \mathbf{x}_2)p(\mathbf{z}_1|\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{z}_1|\mathbf{x}_1)p(\mathbf{z}_2|\mathbf{z}_1, \mathbf{x}_1, \mathbf{x}_2) \end{aligned} \tag{5.2}$$

In the last step we assume independence of $\mathbf{z}_1$ with respect to $\mathbf{x}_2$ and $\mathbf{z}_2$, i.e, $\mathbf{z}_1 \neq \mathbf{z}_1(\mathbf{x}_2, \mathbf{z}_2)$. This formulation allows to simultaneously adapt both features, performing more robustly than the 'independent' case.

MacCormick and Blake (76) and MacCormick and Isard (77) presented a closely related work, that introduces the partitioned sampling, a technique allowing to reduce the curse of dimensionality problem, affecting particle filters dealing with state vectors of high dimensionality. This method applies the 'hypotheses generation' and 'hypothesis correction' stages of a particle filter, separately for different parts of the state vector. However, partitioned sampling is specific for particle filters, and does not allow to integrate other types of algorithms. In this sense, our formulation is more general, since any individual module might be represented by any Bayesian filter or any algorithm with a PDF as output. As we have mentioned in Chapter 4, Wu and Huang (142) propose an approach closely related to the partitioned sampling, where two features (color and shape) are estimated using a single particle filter, but considering separate observations. Again, this framework is restricted to particle filters, and does not allow to integrate other algorithms that might be suitable. Furthermore, in (142), the number of samples necessary to estimate each feature is not optimized, and each time step contains an inner iterative process that increases noticeably the cost of the algorithm.

Another key difference between these 'partitioned sampling' based approaches and the method proposed in this dissertation, is that using our approach it is more probable to associate

[1] The *conditional probability* of an event a assuming the event b is given, denoted by $p(a|b)$, is by definition the ratio $p(a|b) = \frac{p(a,b)}{p(b)}$, where it is assumed that $p(b)$ is not 0.

the best state vector configuration of one feature with the best state vector configuration of another feature, and thus maximizing the join probability. In the works of (76; 77) and (142) this is not guaranteed. This point will be discussed in more detail in following sections.

With these considerations, we can summarize the main contributions of our framework to integrate multiple cues as follows:

1. We propose a probabilistic framework that can integrate as many features as necessary, for tracking purposes. It is worth noting that:

   (a) The state of the features may be approximated by any algorithm which outputs a PDF. In particular, we have integrated features approximated by a Kalman filter with features estimated through particle filters.

   (b) The proposed framework is theoretically proven and validated in a tracking example of synthetically generated data.

   (c) The method allows to integrate both conditionally dependent and conditionally independent cues. In case of feature dependence, the relation between features is considered during the observation phase of the algorithm.

   (d) By estimating each feature with a separated algorithm, the probability of associating the best estimate of one feature with the best estimate of another feature is maximized, and thus the join probability is also maximized.

2. The proposed framework is applied to develop a robust tracking system that simultaneously: **(a)** Adapts the colorspace where image points are represented. **(b)** Updates the distributions of the object and background colorpoints. **(c)** Accommodates the contour of the object.

3. The representation of the color feature by particle filters and the online adaption of the colorspace are novel contributions of our work and make our system capable to track objects in complex and highly cluttered environments, altered by unexpected changes of color and illumination.

The rest of the chapter is organized as follows: Section 5.2 introduces the mathematical framework of the method. In Section 5.3, a comprehensible example for one dimensional cues will be explained, which will be used as a benchmark to compare the performance of our approach

with the partitioned sampling approaches previously mentioned. The state vector of the features used in the real operation of the method (which were described in Chapter 3), and its dynamic models are described in Section 5.4. In Section 5.5 we depict details about the complete tracking algorithm. Results and conclusions are given in Sections 5.6 and 5.7, respectively.

5.2 Mathematical framework

In this Section we will define the mathematical background for the proposed framework. We will start by defining the integration process of conditionally dependent features, and next, we will explain how the conditional dependence between features is considered into the observation model.

5.2.1 Integration process

In the general case, let us describe the object being tracked by a set of F features, whose configuration is specified by the state vectors $\mathbf{x}_1, \ldots, \mathbf{x}_F$, that are sequentially conditionally dependent, i.e., feature i depends on feature $i-1$ (later we will see that the integration of independent cues is straightforward). These features have an associated set of measurements $\mathbf{z}_1, \ldots, \mathbf{z}_F$, where measurement $\mathbf{z}_i$ allows to update the state vector $\mathbf{x}_i$ of the i-th feature. The conditional a posteriori probability $p_1 = p(\mathbf{x}_1|\mathbf{z}_1), \ldots, p_F = p(\mathbf{x}_F|\mathbf{z}_F)$ is estimated using a corresponding Bayesian filter $\mathcal{BF}_1, \ldots, \mathcal{BF}_F$, such as the Kalman filter or the particle filter defined in Chapter 2. For the whole set of variables we assume that the dependence is only in one direction:

$$\left.\begin{array}{l} \mathbf{x}_k = \mathbf{x}_k(\mathbf{z}_i, \mathbf{x}_i) \\ \mathbf{z}_k = \mathbf{z}_k(\mathbf{x}_i, \mathbf{z}_i) \end{array}\right\} \iff i < k \tag{5.3}$$

Considering this dependence relationship we can add extra terms to the a posteriori probability computed for each Bayesian filter. In particular, the expression for the a posteriori probability computed by $\mathcal{BF}_i$ will be $p_i = p(\mathbf{x}_i|\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i)$. Keeping this in mind, next we will prove that the whole a posteriori probability can be computed sequentially, as follows:

$$\begin{aligned} P &= p(\mathbf{X}|\mathbf{Z}) = p(\mathbf{x}_1, \ldots, \mathbf{x}_F|\mathbf{z}_1, \ldots, \mathbf{z}_F) \\ &= p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2) \cdots p(\mathbf{x}_F|\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots \mathbf{z}_F) \\ &= p_1 p_2 \cdots p_F \end{aligned} \tag{5.4}$$

Proof. We will prove this by induction, and applying Bayes' rule (39) and Eq. 5.3:

- Proof for 2 features:

$$\begin{aligned} p(\mathbf{x}_1, \mathbf{x}_2|\mathbf{z}_1, \mathbf{z}_2) &= \frac{p(\mathbf{x}_1, \mathbf{x}_2, \mathbf{z}_1, \mathbf{z}_2)}{p(\mathbf{z}_1, \mathbf{z}_2)} \\ &= \frac{p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)p(\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)}{p(\mathbf{z}_1, \mathbf{z}_2)} \\ &= p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)p(\mathbf{x}_1|\mathbf{z}_1, \mathbf{z}_2) \\ &= p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2) \end{aligned}$$

- For $F-1$ features we assume that

$$\begin{aligned} p(\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}|\mathbf{z}_1, \ldots, \mathbf{z}_{F-1}) &= p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)\cdots \\ &\quad \cdots p(\mathbf{x}_{F-1}|\mathbf{x}_1, \ldots, \mathbf{x}_{F-2}, \mathbf{z}_1, \ldots \mathbf{z}_{F-1}) \quad (5.5) \end{aligned}$$

- Proof for F features:

$$\begin{aligned} p(\mathbf{x}_1, \ldots, \mathbf{x}_F|\mathbf{z}_1, \ldots, \mathbf{z}_F) &= \frac{p(\mathbf{x}_1, \ldots, \mathbf{x}_F, \mathbf{z}_1, \ldots, \mathbf{z}_F)}{p(\mathbf{z}_1, \ldots, \mathbf{z}_F)} \\ &= \frac{p(\mathbf{x}_F|\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots, \mathbf{z}_F)p(\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots, \mathbf{z}_F)}{p(\mathbf{z}_1, \ldots, \mathbf{z}_F)} \\ &= \frac{p(\mathbf{x}_F|\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots, \mathbf{z}_F)p(\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}|\mathbf{z}_1, \ldots, \mathbf{z}_F)p(\mathbf{z}_1, \ldots, \mathbf{z}_F)}{p(\mathbf{z}_1, \ldots, \mathbf{z}_F)} \\ &= p(\mathbf{x}_F|\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots, \mathbf{z}_F)p(\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}|\mathbf{z}_1, \ldots, \mathbf{z}_{F-1}) \end{aligned}$$

$$\text{Eq. 5.5} = p(\mathbf{x}_1|\mathbf{z}_1)p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)\ldots p(\mathbf{x}_F|\mathbf{x}_1, \ldots, \mathbf{x}_{F-1}, \mathbf{z}_1, \ldots \mathbf{z}_F)$$

□

Eq. 5.4 tells us that the whole a posteriori probability density function can be computed sequentially, starting with $\mathcal{BF}_1$ to generate $p(\mathbf{x}_1|\mathbf{z}_1)$ which is then used to estimate $p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{z}_1, \mathbf{z}_2)$ with $\mathcal{BF}_2$, and so on. Note that the inclusion of an extra feature $\mathbf{x}_G$ (with the corresponding measurement vector $\mathbf{z}_G$) independent from the rest, is straighforward. We just need to multiply Eq. 5.4 by the posterior $p(\mathbf{x}_G|\mathbf{z}_G)$.

Until now we have only considered the fusion of several Bayesian filters from the static point of view. But in the iterative performance of the method, $\mathcal{BF}_i$ receives as input at iteration t, the output PDF of its state vector $\mathbf{x}_i$ at the iteration $t-1$. We write the time expanded version of the PDF for $\mathcal{BF}_i$ as

$$p_i^t = p(\mathbf{x}_i^t|\mathbf{x}_1^t, \ldots, \mathbf{x}_{i-1}^t, \mathbf{z}_1^t, \ldots, \mathbf{z}_i^t, p_i^{t-1}) \quad (5.6)$$

The expression for the complete PDF from Eq. 5.4 may be expanded as:

$$\begin{aligned} P^t &= p(\mathbf{X}^t|\mathbf{Z}^t) = p(\mathbf{x}_1^t,\ldots,\mathbf{x}_F^t|\mathbf{z}_1^t,\ldots,\mathbf{z}_F^t) \\ &= p(\mathbf{x}_1^t|\mathbf{z}_1^t,p_1^{t-1})\cdots p(\mathbf{x}_F^t|\mathbf{x}_1^t,\ldots,\mathbf{x}_{F-1}^t,\mathbf{z}_1^t,\ldots\mathbf{z}_F^t,p_F^{t-1}) \\ &= p_1^t p_2^t \cdots p_F^t \end{aligned} \tag{5.7}$$

5.2.2 Introducing cue dependence into the observation model

In this subsection we will explain how the cue dependence is handled in the proposed probabilistic framework.

The dependence between cues comes from the fact that in real tracking algorithms, it is common to evaluate the generated hypotheses about a specific feature, according to the state of another feature. For instance, as we have pointed out in the Introduction of this chapter, contour particle filters use to weigh the predicted contour samples based on the ratio of the number of pixels inside the contour with object color versus the number of pixels outside the contour with background color. Therefore, the contour feature, for this kind of observation model, is dependent on the color feature.

Now, let us assume that feature $\mathbf{x}_2$ (estimated by the Bayesian filter $\mathcal{BF}_2$) depends on feature $\mathbf{x}_1$ (estimated by $\mathcal{BF}_1$). The probability distribution $p(\mathbf{x}_1|\mathbf{z}_1)$, is introduced into the observation model of the feature $\mathbf{x}_2$ according to the following cases:

1. $\mathcal{BF}_1$ is a Kalman filter, i.e, $\mathcal{BF}_1 \equiv \mathcal{KF}_1$:
   The posterior probability estimating the state vector $\mathbf{x}_1$ will be a normal distribution, i.e., $p(\mathbf{x}_1|\mathbf{z}_1) = \mathcal{N}_{\mathbf{x}_1}(\boldsymbol{\mu}_{\mathbf{x}_1}, \boldsymbol{\Sigma}_{\mathbf{x}_1})$. Likewise it is done in particle filters, the normal distribution $p(\mathbf{x}_1|\mathbf{z}_1)$ will be uniformly sampled, and approximated by a discrete set of weighted particles $\{\mathbf{s}_{1j}, \pi_{1j}\}$, $j = 1,\ldots,n_1$, where π_{1j} is the result of evaluating the function $\mathcal{N}_{\mathbf{x}_1}(\boldsymbol{\mu}_{\mathbf{x}_1}, \boldsymbol{\Sigma}_{\mathbf{x}_1})$ for $\mathbf{x}_1 = \mathbf{s}_{1j}$.

   (a) $\mathcal{BF}_2$ is a Kalman filter, i.e, $\mathcal{BF}_2 \equiv \mathcal{KF}_2$:
   In order to evaluate the single hypothesis generated by $\mathcal{KF}_2$, the configuration of the state vector corresponding to the feature '1' will be the mean of $p(\mathbf{x}_1|\mathbf{z}_1)$, that is, $\boldsymbol{\mu}_{\mathbf{x}_1}$.

   (b) $\mathcal{BF}_2$ is a particle filter, i.e, $\mathcal{BF}_2 \equiv \mathcal{PF}_2$:
   Let us call $\{\mathbf{s}_{2j}\}$, $j = 1,\ldots,n_2$ the set of n_2 hypotheses that $\mathcal{PF}_2$ generates about the state vector $\mathbf{x}_2$.

Based on the *deterministic resampling algorithm* detailed in Section 2.3, the set $\{\mathbf{s}_{1j}\}$, $j = 1, \ldots, n_1$ is resampled n_2 times according to the weights π_{1j}, to obtain the new set $\{\tilde{\mathbf{s}}_{1j}\}$, $j = 1, \ldots, n_2$.

Finally, each sample $\mathbf{s}_{2j}$ of feature '2', is evaluated considering that the configuration of feature '1' is $\tilde{\mathbf{s}}_{1j}$. Observe, that the more likely is a configuration of the feature '1', the more times will be used to evaluate feature '2'.

2. $\mathcal{BF}_1$ is a particle filter, i.e, $\mathcal{BF}_1 \equiv \mathcal{PF}_1$:
   In this case, the posterior probability estimating the state vector $\mathbf{x}_1$ will be a probability distribution $p(\mathbf{x}_1|\mathbf{z}_1)$ approximated by a discrete set of weighted particles $\{\mathbf{s}_{1j}, \pi_{1j}\}$, $j = 1, \ldots, n_1$.

   (a) $\mathcal{BF}_2$ is a Kalman filter, i.e, $\mathcal{BF}_2 \equiv \mathcal{KF}_2$:
   To evaluate the single hypothesis generated by $\mathcal{KF}_2$, the configuration of the state vector for feature '1' to be considered, will be the expected value of the set $\{\mathbf{s}_{1j}, \pi_{1j}\}$, $j = 1, \ldots, n_1$, computed as:

$$E(\mathbf{x}_1) = \sum_{j=1}^{n_1} \mathbf{s}_{1j}\pi_{1j}$$

   (b) $\mathcal{BF}_2$ is a particle filter, i.e, $\mathcal{BF}_2 \equiv \mathcal{PF}_2$:
   The procedure is exactly the same than in the case 1b.

In Fig. 5.2 we show an example of how the cue dependence is handled in a case where the two Bayesian filters that are integrated, are a Kalman filter ($\mathcal{KF}_1$) and a particle filter ($\mathcal{PF}_2$). For this example, feature '2' estimated by $\mathcal{PF}_2$ depends on feature '1' estimated by $\mathcal{KF}_1$. During the observation phase of the $\mathcal{PF}_2$, the multiple hypothesis $\{\mathbf{s}_{2j}\}$, $j = 1, \ldots, n_2$ generated in the prediction stage of the filter, need to be weighted according to some external measurement. This measurement, will be performed based on the estimate of feature '1' done by $\mathcal{KF}_1$. For this purpose, the a posteriori PDF approximating $p(\mathbf{x}_1|\mathbf{z}_1)$ is discretized into n_1 weighted particles, $\{\mathbf{s}_{1j}, \pi_{1j}\}$, $j = 1, \ldots, n_1$. Subsequently, this set is resampled n_2 times using a sampling with replacement. A set $\{\tilde{\mathbf{s}}_{1j}\}$, $j = 1, \ldots, n_2$ is obtained. Finally, each sample $\mathbf{s}_{2j}$ of the state vector $\mathbf{x}_2$ is weighted using the configuration of feature '1' represented by the sample $\tilde{\mathbf{s}}_{1j}$.

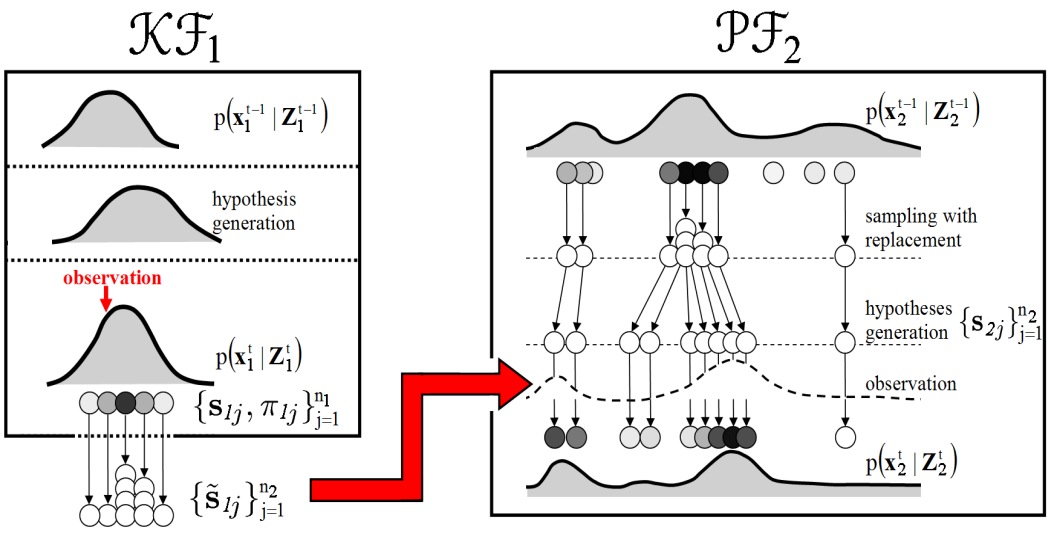


Figure 5.2: **Introducing cue dependence into the observation model.** Example of how cue dependence is handled in the proposed framework, in a case dealing with two features, one estimated by a Kalman filter and the other estimated by a particle filter. The estimate $\mathbf{x}_1$ of feature '1' state vector, carried out by $\mathcal{KF}_1$, is represented by a set of weighted samples $\{\mathbf{s}_{1j}, \pi_{1j}\}$, $j = 1, \ldots, n_1$. These particles are resampled n_2 times (according to their weights), in order to obtain the set $\{\tilde{\mathbf{s}}_{1j}\}$, $j = 1, \ldots, n_2$. Finally, each sample $\mathbf{s}_{2j}$, $j = 1, \ldots, n_2$ of feature '2' state vector, is weighted according to the configuration of the corresponding sample $\tilde{\mathbf{s}}_{1j}$.

Observe in Fig. 5.2 that the samples $\{\mathbf{s}_{1j}\}$ which have higher weights, have more chance to be selected several times when evaluating the hypotheses $\{\mathbf{s}_{2j}\}$; thus allowing to group together the more likely samples of feature '1' with the more likely samples of feature '2'. Also it is important to note that not all the features need to be approximated by the same number of samples. In the example just described, $\mathbf{x_1}$ is estimated by $n_1 = 5$ samples, whereas $\mathbf{x_2}$ is estimated by $n_2 = 10$ samples. This is an important advantage of the proposed framework, especially when dealing with particle filters, since it permits to adapt the number of necessary samples to estimate each feature, as a function of its particular requirements. Some features might require a large number of samples to be appropriately estimated, while other features might require just a reduced number of samples.

To make all the mathematical foundations more clear, in the next section we will apply this method for a simulated case, with only two one-dimensional particle filters.

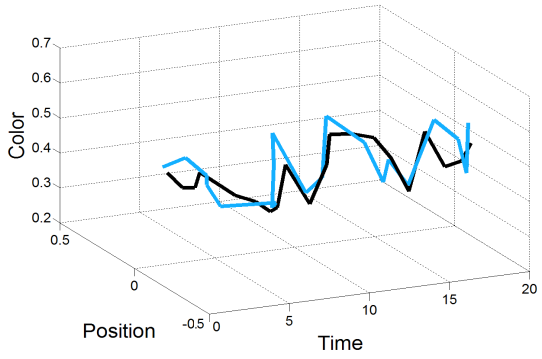


Figure 5.3: **Simulated true (black line) and observed (light blue line) paths described by a point moving on a *color-position* space.** For this example, color and position are represented by one dimensional features.

5.3 Dependent object features in 1D

Let us assume that we want to track a single point that changes its position and color. Both features lie on a one-dimensional space, that is, the point is moving on the horizontal axis, between the $[-1, 1]$ coordinates, and the color is also represented by a single value in the $[0, 1]$ interval. The movement of the point is simulated with a random dynamic model (centered in μ_{pos} and scaled by α_{pos}). Furthermore, we simulate an observation model, adding Gaussian noise to the simulated position :

$$\begin{aligned} \text{pos}^t &= (\text{pos}^{t-1} - \mu_{\text{pos}})\alpha_{\text{pos}} + \mathcal{N}(\mu_{\text{noise,pos}}, \sigma_{\text{noise,pos}}) \\ \text{obs\_pos}^t &= \text{pos}^t + \mathcal{N}(\mu_{\text{noise,obs\_pos}}, \sigma_{\text{noise,obs\_pos}}) \end{aligned} \tag{5.8}$$

Similar equations generate the models for color change and observation:

$$\begin{aligned} \text{col}^t &= (\text{col}^{t-1} - \mu_{\text{col}})\alpha_{\text{col}} + \mathcal{N}(\mu_{\text{noise,col}}, \sigma_{\text{noise,col}}) \\ \text{obs\_col}^t &= \text{col}^t + \mathcal{N}(\mu_{\text{noise,obs\_col}}, \sigma_{\text{noise,obs\_col}}) \end{aligned} \tag{5.9}$$

Figure 5.3 shows the true and observed paths described by the simulated point, in the 'color-position-time' space. Observe that the movement of the point suffers from abrupt changes in both the color and position coordinates. As it is shown in (51), this kind of dynamics can be succesfully tracked using particle filters. Therefore, the state of each one of the features will be estimated through particle filters. We will use $\mathcal{PF}_1$ to track the color, with $\mathbf{x}_1$ and $\mathbf{z}_1$ representing the color state vector and its measurement, and $\mathcal{PF}_2$, $\mathbf{x}_2$ and $\mathbf{z}_2$ the corresponding particle

filter, state vector and measurements assigned to the position. Thus, considering Equations 5.8 and 5.9, we make the following analogies:

$$\begin{array}{lll} \mathcal{PF}_1: & \mathbf{x}_1 = \text{col} & \mathbf{z}_1 = \text{obs\_col} \\ \mathcal{PF}_2: & \mathbf{x}_2 = \text{pos} & \mathbf{z}_2 = \text{obs\_pos} \end{array}$$

At the starting point of iteration t, $\mathcal{PF}_1$ receives at its input p_1^{t-1}, the PDF of the color at time $t-1$, approximated with n_1 weighted samples $\{\mathbf{s}_{1j}^{t-1}, \pi_{1j}^{t-1}\}$, $j = 1, \ldots, n_1$. This set is resampled and propagated according to a random dynamic model of Gaussian noise:

$$\mathbf{s}_{1j}^t = \tilde{\mathbf{s}}_{1j}^{t-1} + \mathcal{N}(0, \sigma_{\text{dyn,col}})$$

where $\tilde{\mathbf{s}}_{1j}^{t-1}$ are the resampled particles.

Each one of these propagated samples is weighted taking into account its proximity to the observed value of the color:

$$\pi_{1j}^t \sim e^{-(\|\mathbf{s}_{1j}^t - \text{obs\_col}^t\|)}$$

The set $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}$, $j = 1, \ldots, n_1$, is the output of the $\mathcal{PF}_1$ and represents an approximation to the a posteriori probability distribution p_1^t. This PDF, jointly with p_2^{t-1} feeds into $\mathcal{PF}_2$, the particle filter responsible for estimating the position of the point. As in the previous particle filter, p_2^{t-1} is approximated by a set of n_2 samples and weights $\{\mathbf{s}_{2j}^{t-1}, \pi_{2j}^{t-1}\}$, $j = 1, \ldots, n_2$, which are resampled and propagated using a random Gaussian dynamic model:

$$\mathbf{s}_{2j}^t = \tilde{\mathbf{s}}_{2j}^{t-1} + \mathcal{N}(0, \sigma_{\text{dyn,pos}})$$

As we have previously pointed out, in real trackers, it is common to evaluate several target positions based on some appearance measure of the object, in our case, color. So we will proceed in a similar way for the weighting stage. To each sample $\mathbf{s}_{2j}^t$, representing a position of the point in space, we associate a sample $\mathbf{s}_{1k}^t$, representing a color state in the color-space, based on the weight π_{1k}^t. This means that those color samples having larger weights (high probability) will be used more times than those having low probability. In order to simulate the weighting of the position samples taking into account the color configuration, the weight assigned to each sample $\mathbf{s}_{2j}^t$ is computed with the following function, which considers both the position and color state vectors:

$$\pi_{2j}^t \sim e^{-(\|\mathbf{s}_{1k}^t - \text{obs\_col}^t\| + \|\mathbf{s}_{2j}^t - \text{obs\_pos}^t\|)}$$

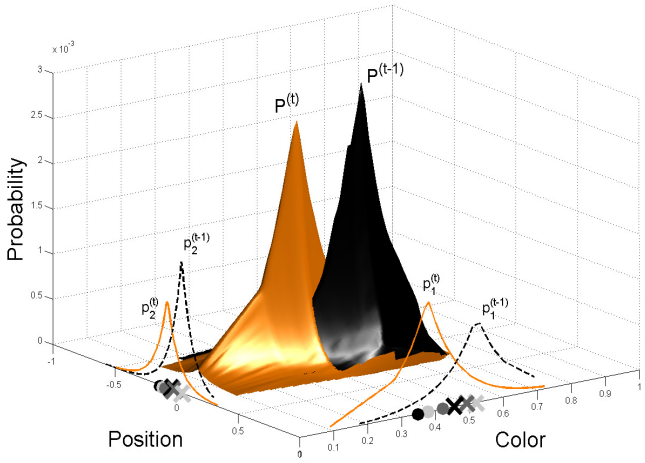


Figure 5.4: **A posteriori PDF's that take part in one iteration of the proposed algorithm.** p_1^{t-1}, p_2^{t-1} and P^{t-1} are the input PDF's and p_1^t, p_2^t and P^t are the output PDF's of the iteration. Crosses represent the data values at time $t-1$ and circles are the data values at time t. The gray level of crosses and circles indicates if the data corresponds to the real value (black), to the estimation done by the filter (dark gray) or the to observation (light gray).

The set $\{\mathbf{s}_{2j}^t, \pi_{2j}^t\}$, $j = 1, \ldots, n_2$, represents the approximation to p_2^t. Finally, we can compute the complete a posteriori probability of the system at time t by:

$$P^t = p(\mathbf{x}_1^t, \mathbf{x}_2^t | \mathbf{z}_1^t, \mathbf{z}_2^t) = p_1^t p_2^t$$

In Fig. 5.4 we show all the a posteriori PDF's in the 'color-position' space, generated in one iteration of the algorithm. The estimates are computed sequentially for each one of the features. First, the color state (p_1^t) is estimated, based on the estimate at previous time step (p_1^{t-1}) and the color observation ($\mathbf{z}_1^t$). Subsequently, in cooperation with p_2^{t-1} and $\mathbf{z}_2^t$, p_1^t is used to estimate the probability distribution of the position feature p_2^t. Finally, the join probability P^t is computed as the product of p_1^t and p_2^t.

5.3.1 Comparison with other approaches

The simple example just presented, will be considered as a testing sequence in order to compare the efficiency of the integration method proposed in this dissertation, with that of previous approaches, specifically, with the conventional Condensation algorithm (51) assuming inde-

pendent cues and the partitioned sampling algorithm (76; 77) assuming dependence in the propagation stage.

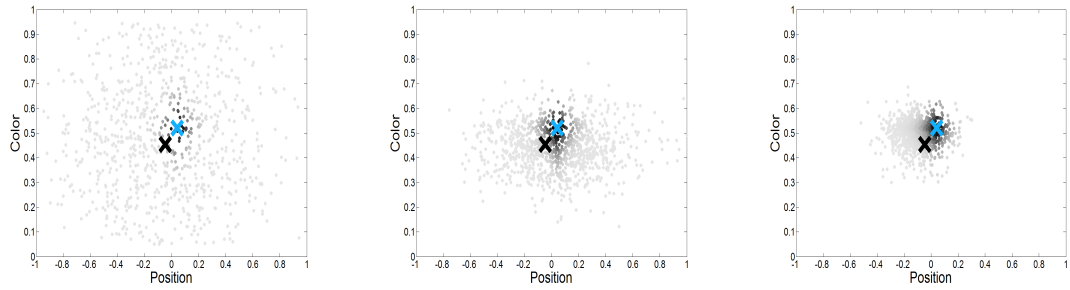
The comparison will be performed in terms of the accuracy in the tracking (distance between the estimated position and color and the true values), and in terms of the *survival diagnostic* (77). The survival diagnostic $\mathcal{D}$ for a particle set $\{\mathbf{s}_i, \pi_i\}$, $i = 1, \ldots, n$ is defined as:

$$\mathcal{D} = \left(\sum_{i=1}^{n} \pi_i^2 \right)^{-1} \tag{5.10}$$

This random variable may be interpreted as the number of particles which would survive a resampling operation, and therefore it is an indicator whether the tracking performance is reliable or not. A low value of $\mathcal{D}$ means that the tracker may lose the target. For instance, if $\pi_1 = 1$ and $\pi_2 = \pi_3 = \ldots = \pi_n = 0$, then $\mathcal{D} = 1$. In these circumstances only one particle might survive the resampling, and tracking would probably fail. On the other hand, if all the particles have the same weight, $\pi_1 = \pi_2 = \ldots = \pi_n = 1/n$ results in that $\mathcal{D} = n$. This indicates that all the n particles would survive an ideal resampling, and the tracking would not get lost. Made this clarification, we proceed to study the performance of different algorithms in the tracking problem proposed in this section.

In the first experiment, the problem has been examined by the conventional Condensation algorithm, assuming that cues are independent. $\mathbf{x}_1$ and $\mathbf{x}_2$ are represented into the same state vector, and the hypotheses generation and correction stages are applied simultaneously to both features. Since the dynamic model of a specific feature has no clue about the state of the other feature, particle samples are spread on a wide area of the state space and, as a consequence, only a few particles will be located in the neighborhood of the true state. Figure 5.5a shows the a posteriori density function obtained in one iteration of the algorithm. The dots represent the different samples (in the 'color-position' configuration space), and the crosses are the true value (black) and observed value (blue). The gray level of the particles is proportional to their likelihood (darker gray levels are more probable particles). Observe that only a small number of particles have a large weight. As a consequence, the survival diagnostic for this approach will have low values.

A better approach may be obtained through the partitioned sampling algorithm. In this case, the dynamics and measurements are not applied simultaneously, but are partitioned into two components. First, the dynamics are applied in the $\mathbf{x}_1$ direction, and therefore the particles are rearranged so that they concentrate around the color observation (by a process called *weighted*



(a) Condensation (b) Partitioned Sampling (c) Proposed Method

Figure 5.5: **A posteriori probability distributions for different particle filter based algorithms.** Comparison of the posterior obtained for three algorithms in the tracking example presented in this section corresponding to a point moving in the 'color-position' space. The results are for a particular iteration, and show how the filters approximate the true value (black cross) based on a set of weighted particles (gray level circles). The gray level is proportional to the probability of the sample, in such a way that darker gray levels indicate more likely samples. Since the true value is only ideally available, the correction of the hypotheses is done based on the observation (blue cross), which we have simulated to be the true value plus a Gaussian noise. In the three experiments, have been used the same number of particles ($n = 1000$) and the same dynamic models. However, note that the approach proposed in this thesis is the method that concentrates a maximum number of samples around the true value.

resampling (76) which keeps the distribution unchanged). This arrangement enhances the estimation by concentrating more particles around the true state. Note in Figure 5.5b this effect on the posterior distribution. Although particles are spread in the $\mathbf{x}_2$ direction, their variability along the $\mathbf{x}_1$ direction is highly reduced. As a result, the number of particles having a large weight is considerably bigger than when using the conventional Condensation.

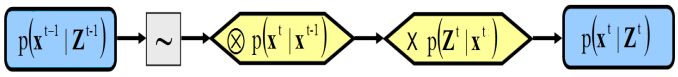
It is important to note that in partitioned sampling, particles are propagated in the direction $\mathbf{x}_2$ according to the likelihood of the samples of feature $\mathbf{x}_1$. Thus, best hypotheses of feature $\mathbf{x}_1$ have more chances to be propagated in the direction $\mathbf{x}_2$. Although this approach outperforms the conventional Condensation algorithm, it still has a limitation, in that the best samples of feature $\mathbf{x}_1$ do not need to be the best samples of feature $\mathbf{x}_2$. Therefore, the common association of the best samples of feature $\mathbf{x}_1$ with the best samples of feature $\mathbf{x}_2$, is not guaranteed.

This is improved in the integration algorithm proposed in the present thesis. The key difference with respect to the previous approaches, is that we assume a different state vector for each feature, and the hypotheses generation and correction stages are also applied separately. In particular, the propagation of the particles for feature $\mathbf{x}_i$, is performed according to the par-

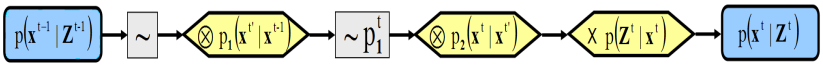
ticles resampling its own probability distribution in the previous time step $p(\mathbf{x}_i^{t-1}|\mathbf{Z}_i^{t-1})$, and not according to the particles that better approximate another feature, avoiding the mentioned problem suffered by partitioned sampling. In Figure 5.5c, we see that proceeding this way the samples are much more concentrated around the true value than they were for the other approaches, what improves noticeably the survival diagnostic.

Furthermore, while partitioned sampling considers the feature dependence during the hypotheses generation stage, we consider it in the hypotheses correction phase, where the posterior of a specific feature is used to weight the samples of another feature. This permits in a same iteration to update all the features representing the target.

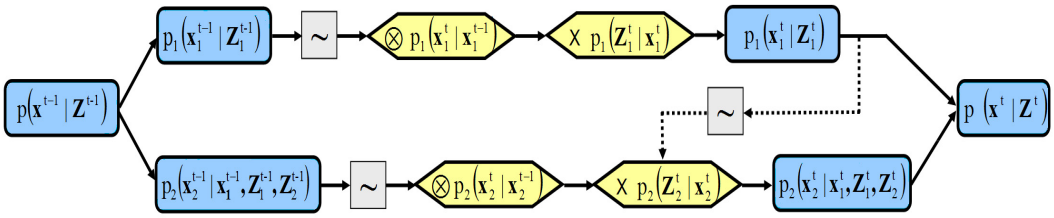
Taking as a model the diagram symbology used in (75) to describe particle filter processes, which we have introduced in Chapter 3, in Figure 5.6 we describe one time step of the conventional Condensation algorithm, the partitioned sampling and the proposed algorithm. These diagrams clearly reflect the difference between the compared algorithms just commented:



(a) Conventional Condensation ($\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]$)

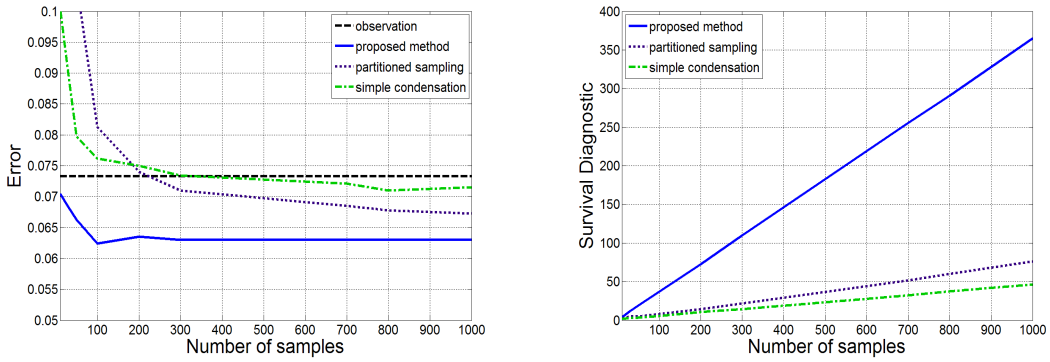


(b) Partitioned Sampling ($\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]$)



(c) Proposed Method

Figure 5.6: **Whole process diagrams of the Conventional Condensation, Partitioned Sampling and the proposed algorithm.** The symbology used in these diagrams is adapted from (75), and it has been presented in Chapter 3 of this thesis. In the partitioned sampling diagram, the symbol $\sim p_1^t$ has been introduced, which indicates a weighted resampling operation with respect to the importance function p_1^t (see (75) for details).



(a) (b)

Figure 5.7: **Tracking results obtained for the conventional Condensation, partitioned sampling and the proposed method.** Analysis of the three algorithms when are applied to the tracking example explained in this section, which was a 20 iterations sequence. The analysis is done in terms of the error (a) in the tracking (distance between the true state and the state estimated by the algorithm) and in terms of the survival rate (b). In both cases the experiments have been realized for different number of samples, and for each specific number of samples, 25 repetitions of the simulation have been done. The results we show, correspond to the mean of these 25 repetitions, of 20 iterations each. Observe that the results agree with the a posteriori distributions plotted in Fig 5.5, as the proposed method outperforms both the Condensation and the partitioned sampling algorithms.

The plots of Figure 5.7 show the tracking results obtained for the three algorithms compared in this section. In Figure 5.7a the algorithms are compared in terms of the tracking error, where the error is computed as the distance between the filter estimate and the true value. For instance, given a posterior approximated by the set $\{\mathbf{s}_j, \pi_j\}$, $j = 1, \ldots, n$, and the true state of the tracked point given by $\mathbf{x}_{true}$, the value of the error is:

$$\mathcal{E}(n) = \|E(\mathbf{x}) - \mathbf{x}_{true}\| \tag{5.11}$$

where, $E(\mathbf{x})$ is the expected value approximated by the filter, i.e, $E(\mathbf{x}) = \sum_{j=1}^{n} \mathbf{s}_j \pi_j$, and $\|\cdot\|$ refers to the Euclidean norm. Observe that the error produced using the method proposed in the dissertation is clearly smaller than the one produced by the other algorithms.

When analyzing the survival diagnostic for the same experiments, we may reach similar conclusions. From Figure 5.7b it can be seen that the largest survival rates, and hence the most reliable tracking results, are obtained when using the integration technique presented in this chapter.

Just a final remark for this section, concerning to the number of particles necessary to achieve a desired level of performance. It is well known that the *curse of dimensionality* is one of the main problems affecting particle filters, that is, when the dimensionality of the state space increases, the number of required samples increases exponentially (60; 76; 142). Intuitively, the number of samples is proportional to the volume of the search space. For instance, if a one-dimensional space is sampled by n particles, the same sampling density in a two dimensional space will require n^2 particles, and so on. Nevertheless, in the proposed method the high dimensional state vectors are separated into various small state vectors and the sampling is particularized for of each low dimensional configuration space. The final number of required particles corresponds to the sum of the particles used in each of these low dimensional spaces. For example, if a two dimensional state vector can be separated into two one-dimensional state vectors, the number of samples may be reduced from n^2 (required in the two dimensional configuration space) to $2n$ (required in the two one-dimensional spaces).

Furthermore, as we have previously pointed out, the number of samples may be adapted for the particular requirements of each of the low dimensional state vectors.

5.4 Feature parameterization and dynamic model

In the preceding sections of this chapter, the integration framework has been presented from a general point of view, and applied to a simple example involving one dimensional features, which has allowed to highlight the important properties of the method, and compare it with other approaches.

The rest of the chapter will describe a particular application of the proposed framework for designing a tracking system able to work in real and dynamic environments. The target is going to be represented by the features described in Chapter 3, which were a bounding box that roughly approximated the image region where the target is expected to be, the color space that best discriminated the target color from the background color, the color distribution into this specific color space, and the object contour. In the following subsections, the parameterization of these features and their respective dynamic models will be described.

5.4.1 Object bounding box

State vector:

The bounding box of the object is just a rectangular shape, that gives a rough estimate about the target position. It is parameterized by the following state vector $\mathbf{x}_1 \in \mathbb{R}_{5\times1}$:

$$\mathbf{x}_1 = [u_1, v_1, a_1, b_1, \theta_1]^T \tag{5.12}$$

where (u_1, v_1) are the coordinates of the center, a_1 and b_1 are the lengths of the sides of the rectangle, and θ_1 is the angle between a_1 and the horizontal axis (see Fig. 5.8).

Dynamic model:

Since the bounding box is just used as a coarse initial estimate of the target position at each iteration, its state may be calculated approximately by a Kalman filter, which mostly relies in the prediction term. It has been stated in Chapter 2 (Section 5) that Kalman filter is based on linear dynamic models with Gaussian white noise. However, a better approximation to the dynamics is achieved by a second order process such as:

$$\mathbf{x}_1^t = \mathbf{x}_1^{t-1} + \alpha(\mathbf{x}_1^{t-1} - \mathbf{x}_1^{t-2}) + \mathbf{q}_{1,h} \tag{5.13}$$

where the parameter α settles the adaptation velocity of the model and $\mathbf{q}_{1,h} \in \mathbb{R}_{5\times1}$ is a random variable distributed as a Gaussian with zero mean and diagonal covariance matrix $\mathbf{\Sigma}_{1,h} \in \mathbb{R}_{5\times5}$.

In order to convert Eq. 5.13 into a linear dynamic model, it is considered an augmented state-vector $\bar{\mathbf{x}}_1^t \in \mathbb{R}_{10\times1}$:

$$\bar{\mathbf{x}}_1^t = \begin{bmatrix} \mathbf{x}_1^{t-1} \\ \mathbf{x}_1^t \end{bmatrix} \tag{5.14}$$

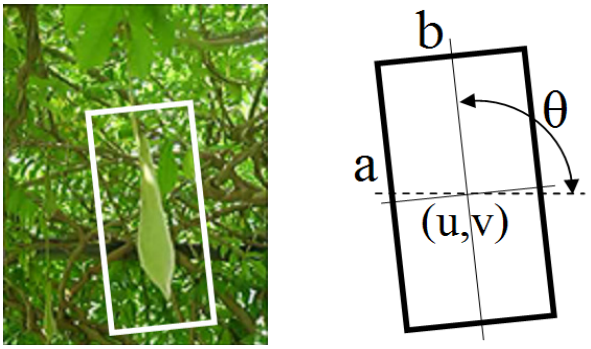
Then, the dynamic model may be rewritten as:

$$\bar{\mathbf{x}}_1^t = \mathbf{H}_1\bar{\mathbf{x}}_1^{t-1} + \mathbf{B}_1\mathbf{q}_{1,h} \tag{5.15}$$

where

$$\mathbf{H}_1 = \begin{bmatrix} \mathbf{0}_{5\times5} & \mathbb{I}_{5\times5} \\ -\alpha\cdot\mathbb{I}_{5\times5} & (1+\alpha)\cdot\mathbb{I}_{5\times5} \end{bmatrix} \quad \text{and} \quad \mathbf{B}_1 = \begin{bmatrix} \mathbf{0}_{5\times5} \\ \mathbb{I}_{5\times5} \end{bmatrix} \tag{5.16}$$

This notation allows to represent a second order process as a linear dynamic model.



(a) (b)

Figure 5.8: **Bounding box feature.** A rough estimation of the object position will be provided by a bounding box surrounding the object, with a gap big enough to ensure that it contains the target. (a) Bounding box example. (b) Bounding box parameterization.

5.4.2 Color space

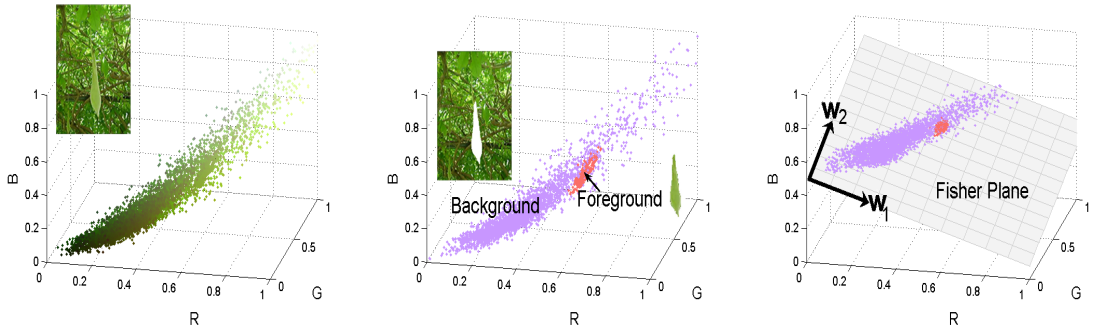
An important contribution of this dissertation is the use of an object dependent colorspace (called Fisher colorspace) that maximizes the distance between the object and background colorpoints, which is a desired property for any color-based tracking system. This kind of colorspace may be understood as an intrinsic object feature adaptable throughout the sequence.

Fisher plane is initialized by a training image, where the RGB colorpoints are manually separated into foreground and background (Figure 5.9a,b). Following the procedure described in Chapter 3 (Section 2), the Fisher plane is determined as the plane that maximizes the distance between the projected classes, while they maintain a low variance (Figure 5.9c). Although it has been proved that the Fisher plane is robust to illumination changes, the movement of the objects in image, or complex illumination effects such as cast shadows, specularities, interreflexions, etc., will cause the color distributions of both the object and background to change. As a consequence, the Fisher plane needs to be adapted online. In particular, the adaptation will be realized through a particle filter, with the following state vector and dynamic model:

State vector:

A Fisher plane spanned by the vectors $\mathbf{w}_1$ and $\mathbf{w}_2$, is parameterized by its normal vector,

$$\mathbf{x}_2 = \frac{\mathbf{w}_1 \times \mathbf{w}_2}{\|\mathbf{w}_1 \times \mathbf{w}_2\|} \in \mathbb{R}_{3\times 1} \tag{5.17}$$



(a) (b) (c)

Figure 5.9: **Fisher colorspace**. (a) Representation of all image points in the *RGB* colorspace. In the upper left corner of the figure the original image is shown. (b) Manual classification of image points into foreground ($\mathcal{O}$) and background ($\mathcal{B}$) classes. The foreground (target to track) is the leaf appearing in the center of the image. (c) The Fisher plane is determined from the training points. This plane maximizes the separation of the projected classes, while keeping a low variance.

Dynamic model:

The Fisher plane will be propagated by a dynamic model, consisting of a random scaling and a random translation:

$$\mathbf{x}_2^t = \mathbf{H}_2 \mathbf{x}_2^{t-1} + \mathbf{q}_2 \tag{5.18}$$

where $\mathbf{H}_2 \sim \mathbf{A}_{3\times3}(\mathbf{0}, \boldsymbol{\sigma}_{\mathbf{H}_2})$ and $\mathbf{q}_2 \sim \mathbf{t}_{3\times1}(\boldsymbol{\mu}_{\mathbf{q}_2}, \boldsymbol{\sigma}_{\mathbf{q}_2})$. The random scaling matrix $\mathbf{A}$ and the random translation vector $\mathbf{t}$ are defined as follows:

$$\mathbf{A}_{m\times m}(\boldsymbol{\mu}_{\mathbf{A}}, \boldsymbol{\sigma}_{\mathbf{A}}) = \begin{bmatrix} 1+a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & 1+a_{mm} \end{bmatrix} \tag{5.19}$$

$$\mathbf{t}_{m\times1}(\boldsymbol{\mu}_{\mathbf{t}}, \boldsymbol{\sigma}_{\mathbf{t}}) = [t_1, \ldots, t_m]^T$$

Variables a_{ij} and t_i are approximated by normal random values, $a_{ij} \sim \mathcal{N}\left(\mu_{a_{ij}}, \sigma_{a_{ij}}\right)$, $t_i \sim \mathcal{N}\left(\mu_{t_i}, \sigma_{t_i}\right)$.

5.4.3 Color distribution

The projection of the *RGB* object and background colorpoints onto the Fisher plane, will be represented by a mixture of Gaussians model. The initial configuration and number of Gaussian components is computed through the *EM* modified algorithm proposed by Figueiredo and

Jain (35), commented in Chapter 3 (Section 3.2). Using this algorithm, the Gaussian components are adjusted independently to the foreground and background colorpoints, represented in the Fisher colorspace (Fig. 5.10a). The state vector and dynamic model for such a model are the following:

State vector:

The configurations of the *MoG* for $\mathcal{O}$ and $\mathcal{B}$ are parameterized by the vector

$$\mathbf{g}_\varepsilon = \begin{bmatrix} \mathbf{p}_\varepsilon \\ \boldsymbol{\mu}_\varepsilon \\ \boldsymbol{\lambda}_\varepsilon \\ \boldsymbol{\theta}_\varepsilon \end{bmatrix} \in \mathbb{R}_{6n_\varepsilon \times 1} \tag{5.20}$$

where $\varepsilon = \{\mathcal{O}, \mathcal{B}\}$, n_ε is the number of Gaussian components for the class ε, $\mathbf{p}_\varepsilon \in \mathbb{R}_{n_\varepsilon \times 1}$ contains the priors for each Gaussian component, $\boldsymbol{\mu}_\varepsilon \in \mathbb{R}_{2n_\varepsilon \times 1}$ the centroids, $\boldsymbol{\lambda}_\varepsilon \in \mathbb{R}_{2n_\varepsilon \times 1}$ the eigenvalues of the principal directions and $\boldsymbol{\theta}_\varepsilon \in \mathbb{R}_{n_\varepsilon \times 1}$ the angles between the principal directions and the horizontal. In Figure 5.10b, all these parameters for a single Gaussian are depicted. The state vector representing the color model will be:

$$\mathbf{x}_3 = \begin{bmatrix} \mathbf{g}_\mathcal{O} \\ \mathbf{g}_\mathcal{B} \end{bmatrix} \in \mathbb{R}_{6n_T \times 1} \tag{5.21}$$

where $n_T = n_\mathcal{O} + n_\mathcal{B}$. Note that the required parameters to completely characterize a Gaussian component are its prior, the centroid and the covariance matrix. The first two components are directly represented in the state vector. The covariance matrix may be deduced from the eigenvalues and angle of the principal component. For instance, let us assume that these parameters are λ_1, λ_2 and θ. The angle θ defines completely the direction of the principal eigenvectors (assumed to be orthogonals), which may be written as $\mathbf{e}_1 = [\cos\theta, \sin\theta]^T$ and $\mathbf{e}_2 = [\sin\theta, -\cos\theta]^T$. Hence, the covariance matrix associated to these parameters will be:

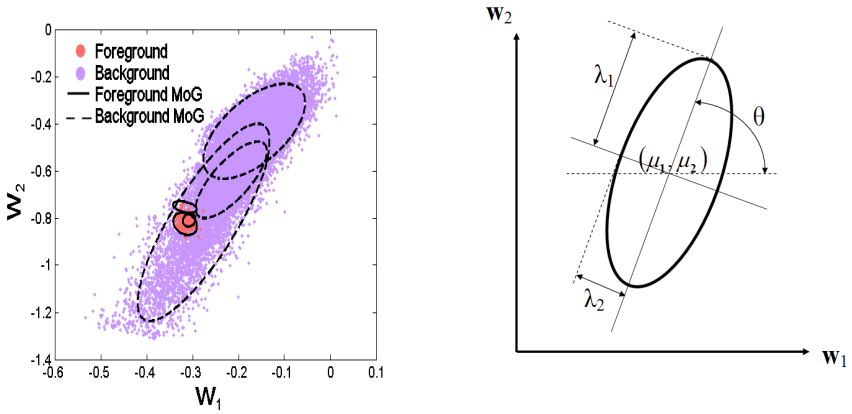
$$\boldsymbol{\Sigma} = [\mathbf{e}_1, \mathbf{e}_2] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} [\mathbf{e}_1, \mathbf{e}_2]^{-1} \tag{5.22}$$

Dynamic model:

Given a parameterization of the state vector $\mathbf{x}_3^{t-1}$ at the previous time step, it will be propagated according to the random dynamic model:

$$\mathbf{x}_3^t = \mathbf{H}_3 \mathbf{x}_3^{t-1} + \mathbf{q}_3 \tag{5.23}$$

where $\mathbf{H}_3 \sim \mathbf{A}_{6n_T \times 6n_T}(\mathbf{0}, \boldsymbol{\sigma}_{\mathbf{H}_3})$, $\mathbf{q}_3 \sim \mathbf{t}_{6n_T \times 1}(\boldsymbol{\mu}_{\mathbf{q}_3}, \boldsymbol{\sigma}_{\mathbf{q}_3})$. Matrix $\mathbf{A}$ and vector $\mathbf{t}$ are defined in Eq. 5.19.



(a) (b)

Figure 5.10: **Color distribution representation**. The set of object and background colorpoints projected onto the Fisher plane, are represented by a *Mixture of Gaussians* (MoG) model. (a) MoG models adjusted to the object and background classes. (b) Detail of the parameters used to represent a single Gaussian component.

5.4.4 Object contour

Since color segmentation usually gives a rough estimation about the target location, we use the contour of the object, to obtain a more precise tracking (see Chapter 3, Section 4). The state vector and dynamic model used to represent the contour are:

State vector:

As stated in Chapter 3 (Sect.4), the contour will be represented by n_c points in the image, $\mathbf{R} = [(u_1, v_1)^T, \ldots, (u_{n_c}, v_{n_c})^T]^T$. We assign these values to the contour state vector:

$$\mathbf{x}_4 = [(u_1, v_1)^T, \ldots, (u_{n_c}, v_{n_c})^T]^T \in \mathbb{R}_{n_c \times 2} \tag{5.24}$$

Dynamic model:

The contour state vector $\mathbf{x}_4^{t-1}$ is propagated according to a dynamic model that produces affine deformed and randomly translated copies of the original contour:

$$\mathbf{x}_4^t = \mathbf{x}_4^{t-1}\mathbf{H}_4 + \mathbf{q}_4 \tag{5.25}$$

Again, $\mathbf{H}_4 \sim \mathbf{A}_{2\times2}(\mathbf{0}, \boldsymbol{\sigma}_{\mathbf{H}_4})$, $\mathbf{q}_4 \sim \mathbf{t}_{2\times1}(\boldsymbol{\mu}_{\mathbf{q}_4}, \boldsymbol{\sigma}_{\mathbf{q}_4})$.

5.4.5 A note about the parameters of the dynamic models

In all the dynamic models defined above there are certain parameters ($\boldsymbol{\Sigma}_{1,h}, \{\boldsymbol{\sigma}_{\mathbf{H}_i}, \boldsymbol{\sigma}_{\mathbf{p}_i}, \boldsymbol{\mu}_{\mathbf{p}_i}\}$, $i = \{2, 3, 4\}$) that control the random performance of the model. Their value will determine

the level of dispersion of the samples in the configuration space, and although they are an important factor to consider when designing the tracker, they do not need to be estimated with high accuracy. In particular, when using particle filters, poor estimates of these parameters may be compensated by selecting a larger number of particles. On the other hand, a Kalman filter might be more sensitive to the value of the covariance matrix $\mathbf{\Sigma}_{1,h}$ defined in the dynamic model, since its prediction is based on a single hypothesis. Nevertheless, in the tracker system explained in this Section, the role of the Kalman filter is to provide a coarse estimate about the bounding box surrounding the target. Therefore, poor estimates of $\mathbf{\Sigma}_{1,h}$ are not such critical.

With these considerations, in the experiments that will be presented at the end of this chapter, the parameters providing the random behaviour to the dynamic models, have been learned off-line from training hand-segmented sequences using a simple least squares technique.

5.5 The complete tracking algorithm

In this Section we will integrate the tools described previously and analyze in detail the complete method for tracking rigid and non-rigid objects in cluttered environments, under changing illumination. Specifically, the target is going to be tracked, using the estimate of the four features just defined: the bounding box (estimated by a Kalman filter $\mathcal{KF}_1$), the Fisher colorspace (estimated through a particle filter $\mathcal{PF}_2$), the color distribution (estimated through $\mathcal{PF}_3$) and the object contour (estimated using $\mathcal{PF}_4$). In the following subsections, the algorithm will be described step by step. For a better understanding of the method, the reader is encouraged to follow the flow diagram in Fig. 5.11.

5.5.1 Input at iteration t

At time t, for the bounding box feature, it is available the mean and covariance parameters from the previous iteration, which estimate its posterior probability p_1^{t-1}. For the rest of $\mathbf{x}_i$ features, $i = \{2, 3, 4\}$, estimated through particle filters, a set of n_i samples $\mathbf{s}_{ij}^{t-1}$, $j = 1, \ldots, n_i$, is available from the previous iteration. The structure of these samples is the same as the corresponding state vector $\mathbf{x}_i$. Each sample has an associated weight π_{ij}^{t-1}. The whole set approximates the a posteriori PDF of the system, $P^{t-1} = p(\mathbf{X}^{t-1}|\mathbf{Z}^{t-1})$ as defined in Eq. 5.7, where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$ contains the state vectors of all the cues utilized to represent the object, and $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4\}$ refers to the observations measured to evaluate the features. Obviously, also available is the input RGB image at time t, denoted by $\mathbf{I}^{\mathrm{RGB},t}$.

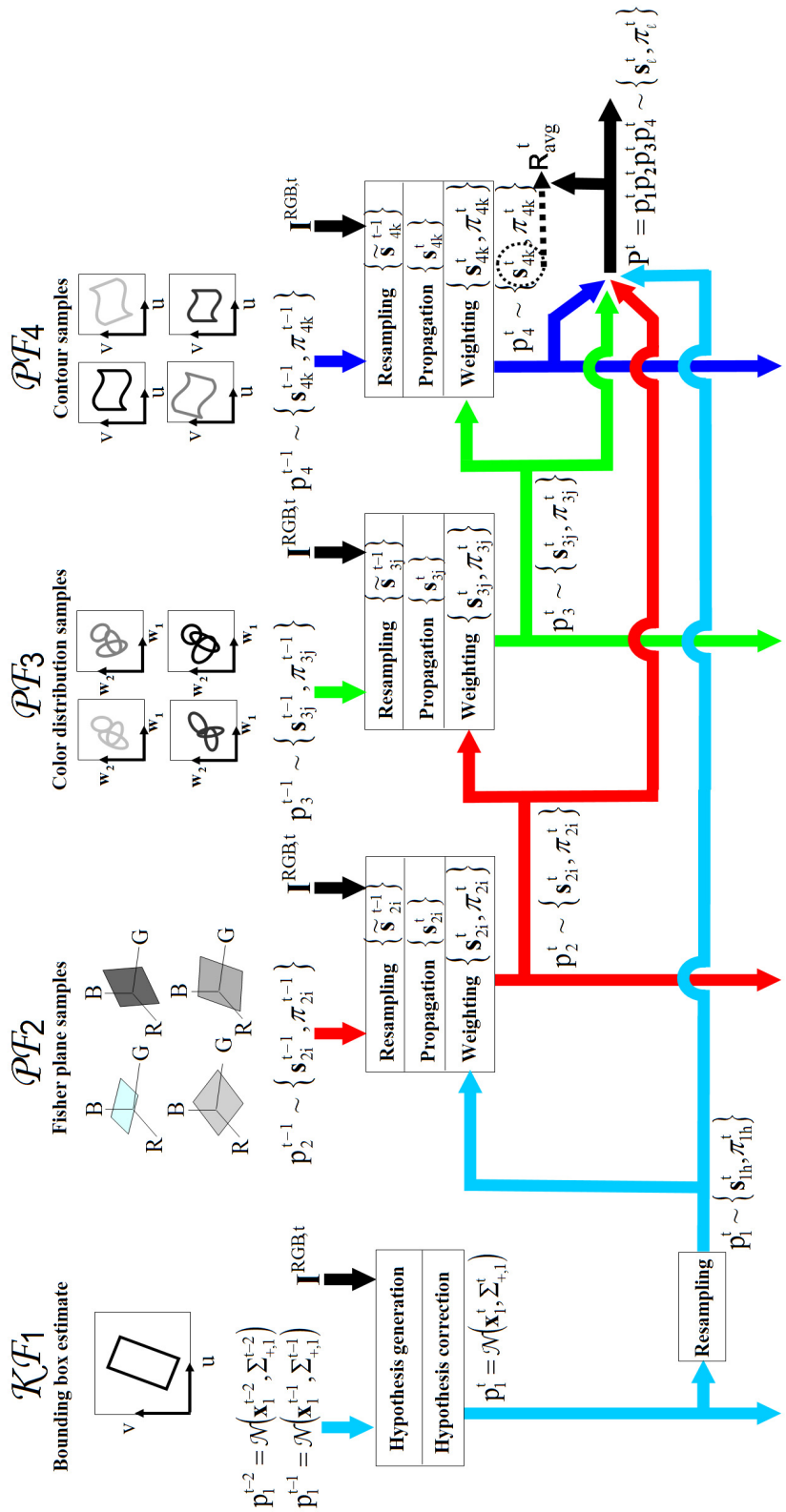


Figure 5.11: **Flow diagram of one iteration of the complete algorithm.** Different color lines and arrows show the paths of each feature. Observe how the output of each filter feeds into a subsequent filter.

5.5.2 Updating the bounding box PDF

It has been previously mentioned that the bounding box is estimated through a Kalman filter, which basically relies in the prediction term of the filter, and the correction introduced by the observation has a low significance. The reason why we do not rely on the bounding box observation is that we wish to deal with highly cluttered sequences, suffering from abrupt illumination changes and unexpected object movements. In these circumstances the observation of a single cue, might probably have a low reliability value. The robustness of the system comes from the integration over all of the cues, and not because of a single cue. Therefore, the estimate of the bounding box state will mostly come from the prediction done by the dynamic model of the filter.

In order to obtain a Kalman filter with such a behaviour, a large value is assigned to the covariance associated with the measurement noise, $\mathbf{\Sigma}_{m,1}^{t}$. Next, let us see in detail how the Kalman filter behaves, under these specifications. The basic steps of the Kalman filter procedure for a single iteration are detailed in Chapter 2 (Section 2), and are repeated here for convenience:

Input data:

$\mathcal{KF}_1$, the Kalman filter associated with state vector $\mathbf{x}_1$, receives the bounding box estimate of the two previous states, i.e, $p_1^{t-1} = \mathcal{N}(\mathbf{x}_1^{t-1}, \mathbf{\Sigma}_1^{t-1})$ and $p_1^{t-2} = \mathcal{N}(\mathbf{x}_1^{t-2}, \mathbf{\Sigma}_1^{t-2})$, where $\mathbf{x}_1$ and $\mathbf{\Sigma}_1$ correspond to the a posteriori estimates of the mean and covariance.

Hypothesis Generation:

As it has been previously argued in Section 5.4.1, the means estimates are arranged into an augmented state vector (in order to implement a second order dynamic model without breaking the Kalman filter assumptions). Similarly, the covariances are arranged into an augmented covariance matrix, which assumes independence between $\mathbf{x}_1^{t-1}$ and $\mathbf{x}_1^{t-2}$:

$$\bar{\mathbf{x}}_{1,+}^{t-1} = \begin{bmatrix} \mathbf{x}_1^{t-2} \\ \mathbf{x}_1^{t-1} \end{bmatrix} \in \mathbb{R}_{10\times 5} \qquad \bar{\mathbf{\Sigma}}_{1,+}^{t-1} = \begin{bmatrix} \mathbf{\Sigma}_1^{t-2} & \mathbf{0}_{5\times 5} \\ \mathbf{0}_{5\times 5} & \mathbf{\Sigma}_1^{t-1} \end{bmatrix} \in \mathbb{R}_{10\times 10}$$

where the subscript symbol '+' indicates that the referred variable is a posterior estimate, and the bar on the top is utilized for the augmented state variables. Using the Kalman filter equations (Eq.2.8 and Eq.2.9), the state vector and covariance matrix are propagated

to:

$$\begin{aligned}\bar{\mathbf{x}}_{1,-}^{t} &= \mathbf{H}_1 \bar{\mathbf{x}}_{1,+}^{t-1} \\ \bar{\mathbf{\Sigma}}_{1,-}^{t} &= \bar{\mathbf{\Sigma}}_{1,d} + \mathbf{H}_1 \bar{\mathbf{\Sigma}}_{1,+}^{t-1} \mathbf{H}_1^T\end{aligned}$$

where the matrix $\mathbf{H}_1 \in \mathbb{R}_{10\times10}$ (Eq. 5.16) takes into account the second order model dynamics, and $\bar{\mathbf{\Sigma}}_{1,d} \in \mathbb{R}_{10\times10}$ is the process noise covariance matrix. Here, the subscript symbol '-' indicates that the estimate is a priori.

Hypothesis Correction:

The first issue to consider in order to perform the hypothesis correction, refers to the observation. Only the part of the state vector concerning to the bounding box parameterization at time t, will be observed, i.e, the *measurement equation* will be expressed as:

$$\mathbf{z}_1^t = \mathbf{M}_1 \bar{\mathbf{x}}_1^t + \mathbf{q}_{1,m}$$

where $\mathbf{M}_1 = [\mathbf{0}_{5\times5}, \mathbb{I}_{5\times5}]$ allows to transform from the 10×1 dimensionality of $\bar{\mathbf{x}}_1^t$ to the 5×1 dimension of the observation $\mathbf{z}_1^t$. The random variable $\mathbf{q}_{1,m}$ stands for the measurement noise, and it is assumed to be zero-mean and Gaussian, $\mathbf{q}_{1,m} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{1,m})$.

In order to determine the state of the observation $\mathbf{z}_1^t$ a simple correlation method is used. Let us call $\mathcal{W}^{t-1}$ the rectangular window defined by the parameters of the state vector $\mathbf{x}_1^{t-1} = \left[u_1^{t-1}, v_1^{t-1}, a_1^{t-1}, b_1^{t-1}, \theta_1^{t-1}\right]^T$. The observation $\mathbf{z}_1^t$, will be the same window but with its centroid translated according to the parameters (du, dv) minimizing the following SSD (*Sum of Squared Differences*) criterion:

$$\arg\min_{du,dv} \left[\sum_{u,v\in\mathcal{W}^{t-1}} \left(\mathbf{I}^{\text{RGB},t-1}(u,v) - \mathbf{I}^{\text{RGB},t}(u+du, v+dv)\right)^2 \right]$$

Subsequently, the value of the observation vector is defined as:

$$\mathbf{z}_1^t = \left[u_1^{t-1} + du, v_1^{t-1} + dv, a_1^{t-1}, b_1^{t-1}, \theta_1^{t-1}\right]^T$$

Nevertheless, this observation is highly sensitive to the presence of clutter or lighting changes, since the SSD operator is not robust under this kind of artifacts. Hence, a low responsibility needs to be assigned to the observation measure about its contribution to the final decision of the a posteriori probability. Kalman filter, allows to control the

relative contribution of the prediction term and the observation term through the values of the dynamic model covariance matrix $\bar{\mathbf{\Sigma}}_{1,h} \in \mathbb{R}_{10\times10}$ and the measurement covariance matrix $\mathbf{\Sigma}_{1,m} \in \mathbb{R}_{5\times5}$. In particular, for the purposes just mentioned, these matrices have been selected offline such that they satisfy:

$$\mathbf{\Sigma}_{1,m} >> \bar{\mathbf{\Sigma}}_{1,h}$$

Actually, since the matrices have a different dimension, this inequality refers to the comparison of $\mathbf{\Sigma}_{1,m}$ and the lower right 5×5 submatrix of $\bar{\mathbf{\Sigma}}_{1,h}$.

Next, let us see (by examining the Kalman hypothesis correction equations) how these constrains on the covariance matrices reduce the contribution of the observation term in the final estimates. From Chapter 2 (Section 2), the hypothesis correction equations may be written as:

$$\begin{aligned}
\mathbf{K}^t &= \bar{\mathbf{\Sigma}}^t_{1,-}(\mathbf{M}^t_1)^T[\mathbf{M}^t_1\bar{\mathbf{\Sigma}}^t_{1,-}(\mathbf{M}^t_1)^T + \mathbf{\Sigma}_{1,m}]^{-1} \\
\bar{\mathbf{x}}^t_{1,+} &= \bar{\mathbf{x}}^t_{1,-} + \mathbf{K}^t[\mathbf{z}^t_1 - \mathbf{M}^t\bar{\mathbf{x}}^t_{1,-}] \\
\bar{\mathbf{\Sigma}}^t_{1,+} &= [\mathbb{I} - \mathbf{K}^t\mathbf{M}^t_1]\bar{\mathbf{\Sigma}}^t_{1,-}
\end{aligned}$$

The first of these equations computes the Kalman gain, $\mathbf{K} \in \mathbb{R}_{5\times10}$. Note that a large measurement covariance matrix implies a small Kalman gain, i.e, $\mathbf{\Sigma}_{1,m} \uparrow\uparrow \Rightarrow \mathbf{K} \downarrow\downarrow$. As a consequence (from the second and third equations):

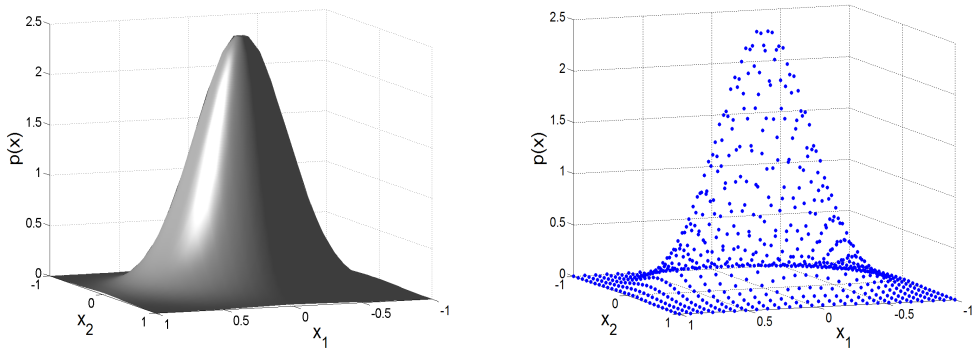
$$\begin{aligned}
\bar{\mathbf{x}}^t_{1,+} &= \bar{\mathbf{x}}^t_{1,-} + \Delta\mathbf{x}^t_1 \\
\bar{\mathbf{\Sigma}}^t_{1,+} &= \bar{\mathbf{\Sigma}}^t_{1,-} + \Delta\mathbf{\Sigma}^t_1
\end{aligned}$$

where $\Delta\mathbf{x}^t_1$ and $\Delta\mathbf{\Sigma}^t_1$ are the corrections introduced by the observations, and are small in comparison with the predicted values $\bar{\mathbf{x}}^t_{1,-}$ and $\bar{\mathbf{\Sigma}}^t_{1,-}$, respectively. Thus, the observations are just utilized as a small bias for the predictions.

Output data:

Uniquely the parts of the augmented state vector and augmented covariance matrix which make reference to the variable $\mathbf{x}^t_1$ are considered in the output of the algorithm:

$$\begin{aligned}
\mathbf{x}^t_1 &= [\bar{\mathbf{x}}^t_{1,+}(6), \ldots, \bar{\mathbf{x}}^t_{1,+}(10)]^T \in \mathbb{R}_{5\times1} \\
\mathbf{\Sigma}^t_1 &= \begin{bmatrix} \bar{\mathbf{\Sigma}}^t_{1,+}(6,6) & \ldots & \bar{\mathbf{\Sigma}}^t_{1,+}(6,6) \\ \vdots & \ddots & \vdots \\ \bar{\mathbf{\Sigma}}^t_{1,+}(10,6) & \ldots & \bar{\mathbf{\Sigma}}^t_{1,+}(10,10) \end{bmatrix} \in \mathbb{R}_{5\times5}
\end{aligned} \tag{5.26}$$



(a) (b)

Figure 5.12: **Uniform sampling of a normal distribution.** (a) Original normal density PDF. (b) Uniformly sampled density.

which determine the a posteriori estimate of the bounding box, $p_1^t = \mathcal{N}(\mathbf{x}_1^t, \mathbf{\Sigma}_1^t)$. Since this distribution is going to feed into subsequent particle filters based on discrete and weighted samples of the state vector, it is necessary to discretize p_1^t. Thus, the normal density p_1^t is uniformly sampled (see Fig. 5.12) and approximated by a set of n_1 weighted particles:

$$p_1^t = \mathcal{N}(\mathbf{x}_1^t, \mathbf{\Sigma}_1^t) \cong \sum_{j=1}^{n_1} \mathbf{s}_{1j}\pi_{1j} \tag{5.27}$$

5.5.3 Updating the Fisher plane PDF

Whereas the bounding box feature is approximately estimated through a kalman filter mostly relying on its prediction component, the rest of the object cues are going to be estimated through particle filters. In this subsection, the particle filter responsible of the Fisher plane feature, $\mathcal{PF}_2$, will be described:

Input data:

At the starting point of iteration t, $\mathcal{PF}_2$, the particle filter associated with $\mathbf{x}_2$, receives p_2^{t-1}, the PDF of the state vector $\mathbf{x}_2$ at time $t-1$, approximated by n_2 weighted samples $\{\mathbf{s}_{2j}^{t-1}, \pi_{2j}^{t-1}\}$, $j = 1, \ldots, n_2$.

In addition, it also receives the output of the previous filter, $\mathcal{KF}_1$ estimating the feature $\mathbf{x}_1$ by a set of n_1 weighted samples, $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}$, $j = 1, \ldots, n_1$.

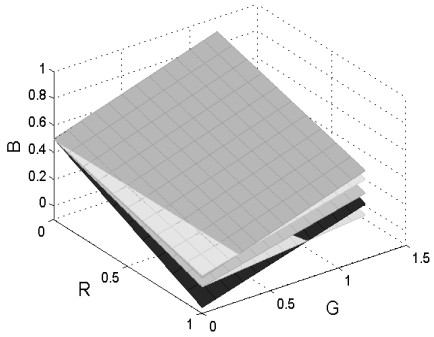


Figure 5.13: **Generation of multiple hypotheses for the Fisher plane feature.**

Hypotheses generation:

Using the standard procedure of the particle filters, the set of particles $\{\mathbf{s}_{2j}^{t-1}, \pi_{2j}^{t-1}\}$, $j = 1, \ldots, n_2$ are resampled using the *deterministic sampling algorithm* (Section 2.3) and propagated to the set $\{\mathbf{s}_{2j}^{t}\}$ according to the dynamic model defined in Section 5.4.2. Each sample represents a different configuration of the Fisher plane, $\mathbf{W}_j$, $j = 1, \ldots, n_2$. Figure 5.13 shows some samples of Fisher planes obtained after the hypotheses generation stage.

Hypotheses correction:

The keypoint of the proposed approach is that the cue dependence is considered during the hypotheses correction stage. In particular, in order to assign a weight to the propagated samples $\{\mathbf{s}_{2j}^{t}\}$, $j = 1, \ldots, n_2$, the information provided from the output p_1^t of $\mathcal{KF}_1$ is used. The discretized samples $\{\mathbf{s}_{1j}^{t}, \pi_{1j}^{t}\}$, $j = 1, \ldots, n_1$ approximating p_1^t are resampled n_2 times, resulting in the set $\{\tilde{\mathbf{s}}_{1j}^{t}\}$, $j = 1, \ldots, n_2$. Note that this set may contain repeated copies of the more likely samples of the bounding box. Then, every Fisher plane sample $\mathbf{s}_{2j}^{t}$ is associated with a bounding box sample $\tilde{\mathbf{s}}_{1j}^{t}$. Let us call $\mathcal{W}_j^t$ the rectangular bounding box defined by $\tilde{\mathbf{s}}_{1j}^{t}$.

Once we have defined a bounding box $\mathcal{W}_j^t$ for each Fisher plane $\mathbf{s}_{2j}^{t}$, the basic idea is to weigh the latter depending on how well it permits to discriminate the points inside $\mathcal{W}_j^t$ from the points outside $\mathcal{W}_j^t$.

To this end we select randomly two sets of RGB colorpoints, $\mathbf{C}_{\mathcal{W}}^{\mathrm{RGB}}$ and $\mathbf{C}_{\overline{\mathcal{W}}}^{\mathrm{RGB}}$, inside and outside $\mathcal{W}_j^t$, respectively. These sets and the image $\mathbf{I}^{\mathrm{RGB},t}$ are projected onto the n_j

Fisher planes, generating the n_j triplets $\{\mathbf{C}^{\text{Fisher}}_{\mathcal{W},j}, \mathbf{C}^{\text{Fisher}}_{\overline{\mathcal{W}},j}, \mathbf{I}^{\text{Fisher},t}_{j}\}$. For each triplet we use the *EM* algorithm to fit a *MoG* to the sets $\mathbf{C}^{\text{Fisher}}_{\mathcal{W},j}$ and $\mathbf{C}^{\text{Fisher}}_{\overline{\mathcal{W}},j}$.

Based on these *MoG*s we compute the a posteriori probability map $p(\mathcal{W}^t_j|\mathbf{I}^{\text{Fisher},t}_j)$ for all the (u,v) pixels of image $\mathbf{I}^{\text{Fisher},t}_j$, using the Bayes rule (Eq. 3.30). According to this probability map, we assign the following weight to each sample:

$$\pi^t_{2j} \sim \frac{\sum_{(u,v)\in\mathcal{W}^t_j} p\left(\mathcal{W}^t_j|\mathbf{I}^{\text{Fisher},t}_j\right)}{n_{\mathcal{W}}} - \frac{\sum_{(u,v)\notin\mathcal{W}^t_j} p\left(\mathcal{W}^t_j|\mathbf{I}^{\text{Fisher},t}_j\right)}{n_{\overline{\mathcal{W}}}}$$

where $n_{\mathcal{W}}$ and $n_{\overline{\mathcal{W}}}$ are the number of image pixels in and out of $\mathcal{W}^t_j$, respectively.

Output data:

The output of $\mathcal{PF}_2$ is the set $\{\mathbf{s}^t_{2j}, \pi^t_{2j}\}$, $j = 1,\ldots,n_2$ approximating the estimate of the a posteriori probability function p^t_2 for the normal to the Fisher plane.

5.5.4 Updating the foreground and background color distributions PDF's

Input data:

$\mathcal{PF}_3$, the particle filter associated with the state vector $\mathbf{x}_3$, receives at its input $p^{t-1}_3 \sim \{\mathbf{s}^{t-1}_{3j}, \pi^{t-1}_{3j}\}$, $j = 1,\ldots,n_3$, approximating the PDF of the color distributions in the previous iteration, and $p^t_2 \sim \{\mathbf{s}^t_{2j}, \pi^t_{1j}\}$, $j = 1,\ldots,n_2$, an approximation to the PDF of the Fisher planes at time t.

Hypotheses generation:

Particles $\{\mathbf{s}^{t-1}_{3j}\}$ are resampled and propagated (using the dynamic model associated with $\mathbf{x}_3$, described in Section 5.4.3) to the set $\{\mathbf{s}^t_{3j}\}$, $j = 1,\ldots,n_3$. A sample $\mathbf{s}^t_{3j}$ represents a *MoG* configuration for the foreground and background color points projected onto the Fisher colorspace. Figure 5.14 (top) shows the appearance of different *MoG*s configurations, resulting from the random propagation generated by the dynamic model.

Hypotheses correction:

Again, in order to assign the weight to these samples we use the information provided from the output p^t_2 of $\mathcal{PF}_2$. By the *deterministic resampling* method described previously, the set $\{\mathbf{s}^t_{2j}\}$, $j = 1,\ldots,n_2$ is resampled n_3 times, providing the set $\{\tilde{\mathbf{s}}^t_{2j}\}$, $j = 1,\ldots,n_3$. This allows to assign the most likely samples $\mathbf{s}^t_{2j}$ of Fisher planes to the samples $\mathbf{s}^t_{3j}$ of *MoG*s.

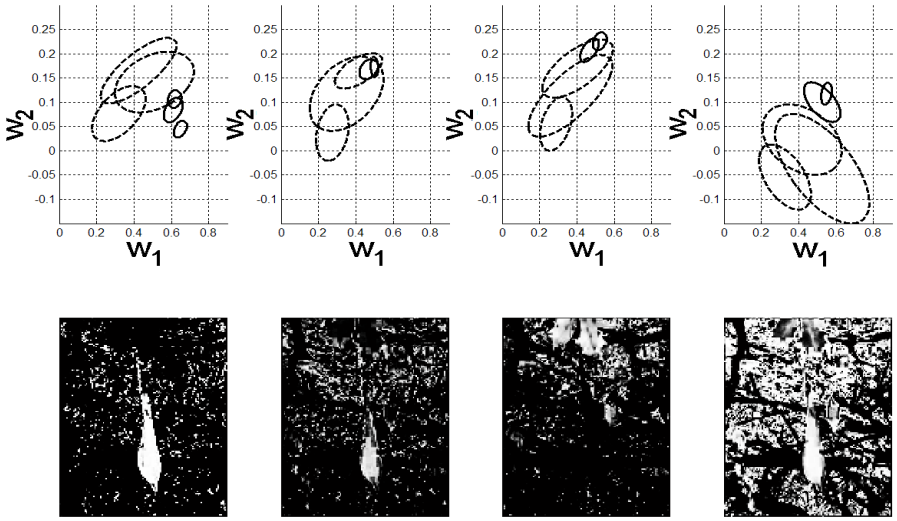


Figure 5.14: **Generation of multiple hypotheses for the foreground ($\mathcal{O}$) and background ($\mathcal{B}$) color distributions**. Top: Several hypothesized *MoG*'s parameterizing the $\mathcal{O}$ and $\mathcal{B}$ color distributions. Solid line ellipses and dashed line ellipses belong to the foreground and background *MoG*'s, respectively. Bottom: A posteriori probability maps of the object class, obtained using the different color configurations. Note that some of the color configurations are appropriate to discriminate the target (central leaf) from the rest of the background, whereas using other configurations, $\mathcal{O}$ and $\mathcal{B}$ regions are undistinguishable.

The rest of the weighting process is similar to the one described in the previous section: for a given sample $\mathbf{s}_{3j}^t$, $j = 1, \ldots, n_3$, we project image $\mathbf{I}^{\text{RGB},t}$ to its associated Fisher plane $\mathbf{W}_j$ parameterized by $\tilde{\mathbf{s}}_{2j}^t$. The new image will be $\mathbf{I}_j^{\text{Fisher},t} = \mathbf{I}^{\text{RGB},t}\mathbf{W}_j^T$.

Using the *MoG*s of the object and background parameterized by the sample $\mathbf{s}_{3j}^t$, the a posteriori probability map $p(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t})$ is computed for all the pixels of $\mathbf{I}_j^{\text{Fisher},t}$, and the weight π_{3j}^t is assigned by:

$$\pi_{3j}^t \sim \frac{\sum_{(u,v)\in\mathcal{W}_j^t} p\left(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t}\right)}{n_{\mathcal{W}}} - \frac{\sum_{(u,v)\notin\mathcal{W}} p\left(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t}\right)}{n_{\overline{\mathcal{W}}}}$$

where $\mathcal{W}_j^t$, $n_{\mathcal{W}}$ and $n_{\overline{\mathcal{W}}}$ were defined above.

In Fig. 5.14 (bottom), the a posteriori probability maps of the target (the central leaf) are depicted. Notice how some of the MoG's configurations provide a probability map where the target is clearly distinguished from the background.

Output data:

The set $\{\mathbf{s}_{3j}^t, \pi_{3j}^t\}$, $j = 1, \ldots, n_3$ approximates the estimate of the a posteriori probability function p_3^t for the fore/background color distributions.

5.5.3 Updating the contour PDF

Input data:

The last particle filter, $\mathcal{PF}_4$, receives at its input $p_4^{t-1} \sim \{\mathbf{s}_{4j}^{t-1}, \pi_{4j}^{(t-1)}\}$, $j = 1, \ldots, n_4$, that approximates the PDF of the contours in the previous iteration, and $p_3^t \sim \{\mathbf{s}_{3j}^t, \pi_{3j}^t\}$, $j = 1, \ldots, n_3$, an approximation to the PDF of the color distributions of foreground and background at time t.

Hypotheses generation:

In the same manner to the procedure utilized in $\mathcal{PF}_2$ and $\mathcal{PF}_3$, particles $\{\mathbf{s}_{4j}^{t-1}\}$ are resampled and propagated to the set $\{\mathbf{s}_{4j}^t\}$, $j = 1, \ldots, n_4$ according to the dynamic model described in Section 5.4.4. This dynamic model, produces affine deformed and translated copies of the original contours (see some examples in Figure 5.15, for the leaf tracking example used in the whole chapter).

Hypotheses correction:

The set $\{\mathbf{s}_{4j}^t\}$ is weighted based on p_3^t through a similar process to the one described for $\mathcal{PF}_2$ and $\mathcal{PF}_3$: initially, samples $\{\mathbf{s}_{3j}^t, \pi_{3j}^t\}$, $j = 1, \ldots, n_3$ are resampled according to the weights π_{3j}^t, resulting in a new set $\{\tilde{\mathbf{s}}_{3j}^t\}$, $j = 1, \ldots, n_4$. Then, every color sample $\tilde{\mathbf{s}}_{3j}^t$, $j = 1, \ldots, n_4$ is associated with each contour sample $\mathbf{s}_{4j}^t$.

The a posteriori probability map $p(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t})$ assigned to $\tilde{\mathbf{s}}_{3j}^t$ in the previous time step, and the contour $\mathbf{R}_j$ represented by $\mathbf{s}_{4j}^t$, are used to compute the weights for the contour samples as follows:

$$\pi_{4j}^t \sim \frac{\sum_{(u,v)\in\mathbf{r}_j} p\left(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t}\right)}{n_{\mathbf{R}_j}} - \frac{\sum_{(u,v)\notin\mathbf{R}_j} p\left(\mathcal{O}|\mathbf{I}_j^{\text{Fisher},t}\right)}{n_{\overline{\mathbf{R}_j}}}$$

where $n_{\mathbf{R}_j}$ and $n_{\overline{\mathbf{R}_j}}$ are the number of image pixels inside and outside the contour $\mathbf{R}_j$.

Output data:

Finally, the set of samples and weights $\{\mathbf{s}_{4j}^t, \pi_{4j}^t\}$, $j = 1, \ldots, n_4$ approximate the estimate of the a posteriori probability function p_4^t for the contours of the object.

Figure 5.15: **Generation of multiple hypotheses for the contour feature.**

5.5.6 Algorithm output generation

As we have shown in Section 5.2, the complete a posteriori probability function, can be determined by

$$\begin{aligned} P^t &= p(\mathbf{x}_1^t, \mathbf{x}_2^t, \mathbf{x}_3^t, \mathbf{x}_4^t | \mathbf{z}_1^t, \mathbf{z}_2^t, \mathbf{z}_3^t, \mathbf{z}_4^t) \\ &= p_1^t p_2^t p_3^t p_4^t \\ &= p_1(\mathbf{x}_1^t|\mathbf{z}_1^t) p_2(\mathbf{x}_2^t|\mathbf{x}_1^t, \mathbf{z}_1^t, \mathbf{z}_2^t) p_3(\mathbf{x}_3^t|\mathbf{x}_1^t, \mathbf{x}_2^t, \mathbf{z}_1^t, \mathbf{z}_2^t, \mathbf{z}_3^t) p_4(\mathbf{x}_4^t|\mathbf{x}_1^t, \mathbf{x}_2^t, \mathbf{x}_3^t, \mathbf{z}_1^t, \mathbf{z}_2^t, \mathbf{z}_3^t, \mathbf{z}_4^t) \\ &\sim \left\{ \left\{ \mathbf{s}_{4k}^t \left(\mathbf{s}_{3j}^t \left(\mathbf{s}_{2i}^t (\mathbf{s}_{1h}^t) \right) \right) \right\}, \left\{ \pi_{1h}^t \pi_{2i}^t \pi_{3j}^t \pi_{4k}^t \right\} \right\} = \{\mathbf{s}_l^t, \pi_l^t\} \end{aligned} \tag{5.28}$$

where $l = 1, \ldots, n_4$. Eq. 5.28 reflects the fact that samples of state vector $\mathbf{x}_4$ are computed taking into account samples of the state vector $\mathbf{x}_3$ (i.e, $\mathbf{s}_{4k}^t \equiv \mathbf{s}_{4k}^t(\mathbf{s}_{3j}^t)$) and these have been computed considering samples of $\mathbf{x}_2$ (i.e, $\mathbf{s}_{3j}^t \equiv \mathbf{s}_{3j}^t(\mathbf{s}_{2i}^t)$) and these have considered samples of $\mathbf{x}_1$ (i.e, $\mathbf{s}_{2i}^t \equiv \mathbf{s}_{2i}^t(\mathbf{s}_{1h}^t)$). Observe that the final number of samples to approximate the whole probability P^t is determined by n_4. Considering the final weights, the average contour is computed as

$$\mathbf{R}_{avg}^t = \sum_{l=1}^{n_4} \mathbf{s}_{4l}^t \pi_l^t \tag{5.29}$$

Since all the contour samples have been constructed with an affine deformation model, we need to add an extra final stage in order to cope with non-linear deformations of the object boundary. We use $\mathbf{R}_{avg}^t$ to initialize a deformable contour that is fitted to the contours of the object using the traditional *snake* formulation (59) detailed in Chapter 3 (Section 4). This adjustment is highly simplified by using the target position estimated by the color particle filter,

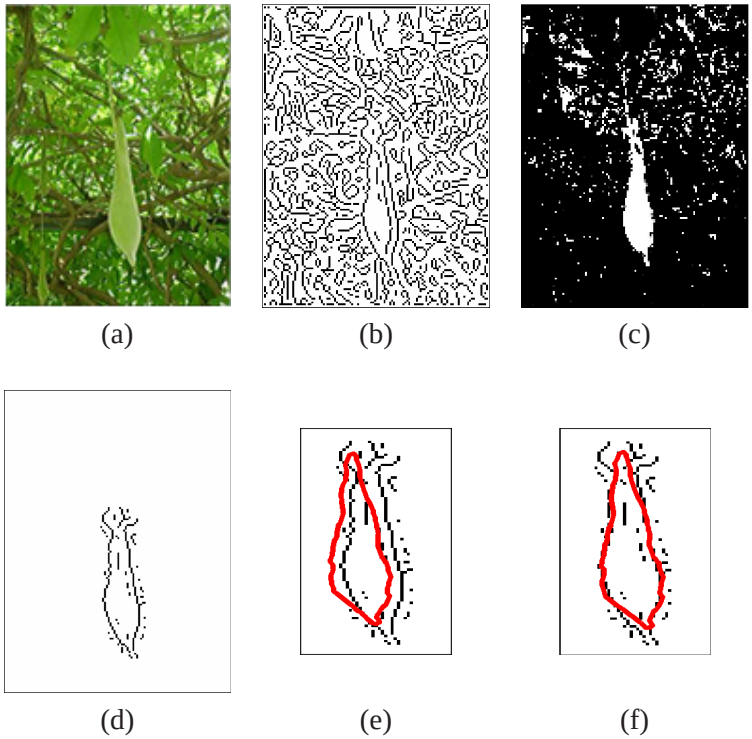


Figure 5.16: **Simplification of the snake fitting procedure using color information.** (a) Original cluttered image. (b) Edge features image obtained with a Canny edge detector. Observe the large quantity of noisy edges detected, which might disrupt a traditional snake procedure from converging to the true object contour. (c) Foreground a posteriori probability map obtained using the color module. (d) Refined edge image, where most of the noisy edges have been removed considering a mask obtained by applying simple morphological operations on image (c). (e) Contour $\mathbf{R}_{avg}^{t}$ used as initialization for a snake fitting procedure . (f) Results of the snake fitting.

as it is shown in Fig. 5.16, where the a posteriori probability map of the color module allows to eliminate noisy edges from the original image.

Note the advantage of using the color module: traditional snake algorithms need to adjust a given curve to the edges of an image. However, if the image contains a high level of clutter (such as the image shown in Fig 5.16a), a standard edge detector may detect a lot of noisy edges which might disturb the snake during the fitting procedure. For instance, Fig 5.16b shows the edges detected by a Canny filter in the previous image. Under this type of edge images, traditional snake algorithms are prone to fail. Nevertheless, by applying simple morphological operations on the a posteriori probability map of the target provided by the color module

(Fig 5.16c) most of the noisy edges may be eliminated from the image (Fig 5.16d). Then, the fitting procedure is considerably made easier. Figures 5.16e and 5.16f show the initialization of the snake (by the averaged contour $\mathbf{R}^t_{avg}$) and the final result of the adjustment, respectively.

5.6 Experimental results

In this Section we present the results of different experiments on both synthetic and real video sequences, and examine the robustness of our system to several changing conditions of the environment, in situations where other algorithms may fail.

Before discussing the results obtained, we would like to point out that since the proposed algorithm has been implemented in an interpretative language (`MATLAB`), we cannot directly discuss the time performance issues. Nevertheless, what is important to note, is that in the particular case of integrating several particle filters, the structure of the fusion framework allows to reduce considerably the number of samples necessary to approximate the PDF representing the state of the target. As we have previously argued in Section 5.3.1, this feature, settles the problem of *curse of dimensionality*, undergone by particle filters when the size of the state vector is increased.

In the following subsections, some experimental results will be reported. The first set of experiments, deal with sequences where the lighting conditions or the appearance of the target change continuously. In the last group of experiments, abrupt illumination changes will be considered. In both cases, there are included examples of targets which deform rigidly and non-rigidly.

5.6.1 Tracking under continuous lighting changes

In the first experiment, it is tracked a synthetically generated sequence of an ellipse that randomly changes its position, color and shape in a cluttered background. In Fig. 5.17(top) we depict the path followed by the color cue. Observe the non-linearity of the trajectory. As it was shown in (51) these kind of paths can not be estimated by filters based on smooth dynamic models, but instead we need to use filters able to cope with the non-linearities, such as the multihypotheses framework offered by particle filters. Results show that the method proposed in this dissertation, based on multiple-multihypotheses algorithms allows to segment and track the ellipse, even when the background has a similar color to the target (observe the frame before last).

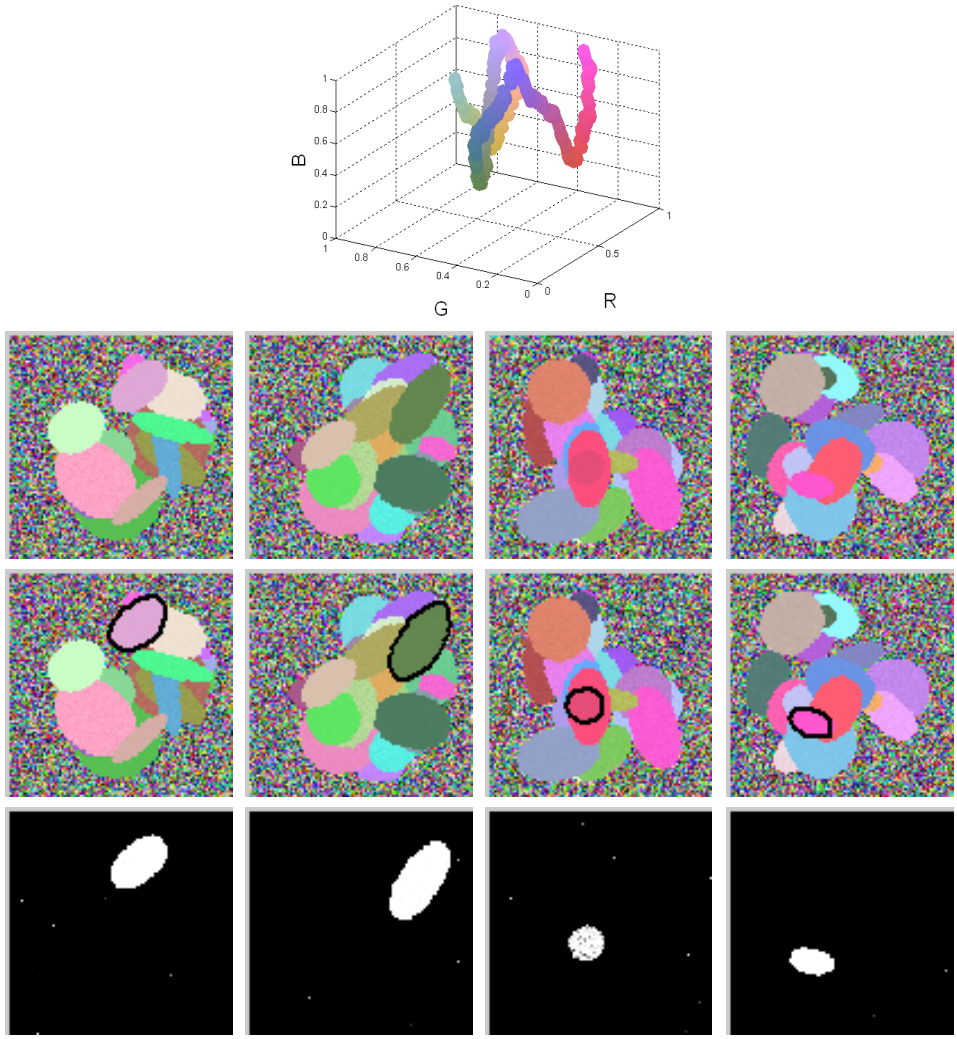


Figure 5.17: **Experiment 1: Tracking of a synthetic ellipse that randomly changes its color, position and shape.** Top: Path followed by the color distribution of the tracked ellipse. The data is represented in the *RGB* colorspace. Note the non-linear change of the color, which can not be predicted by a smooth dynamic model. Bottom: Some sequence frames showing the tracking results; original frames (first row), tracking results (second row), and the target a posteriori PDF map of the color module (third row). The proposed method integrating position prediction, optimal color space selection, color distribution estimate and contour estimate is able to segment the tracked ellipse even when the background contains highly disturbing elements. Observe in the last bust one frame, how the tracked ellipse is surrounded by another ellipse with similar appearance. In spite of that, the tracker does not lose the target.

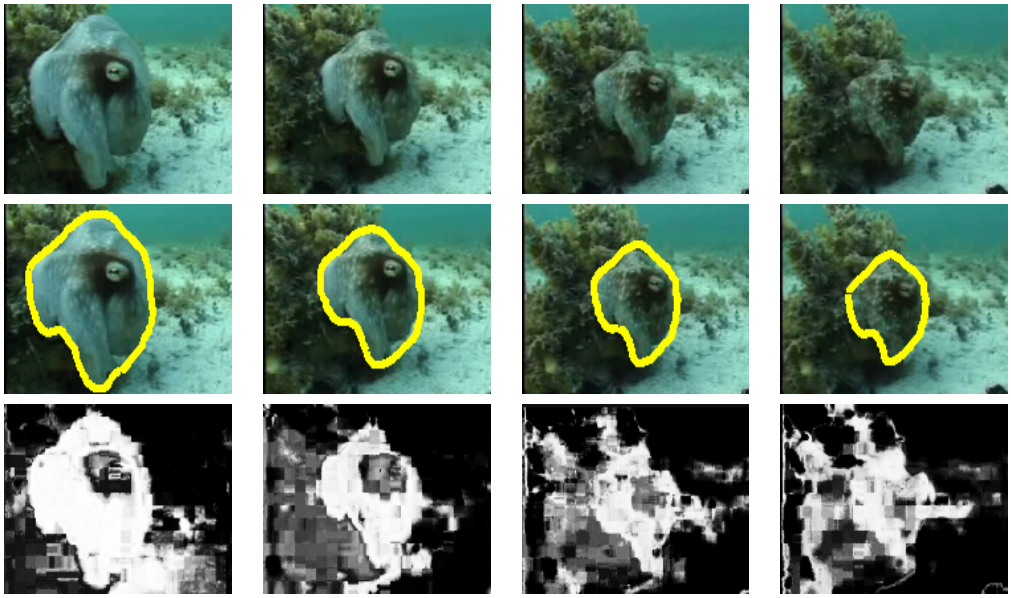


Figure 5.18: **Experiment 2: Tracking a camouflaging octopus.** Top row: Original sequence. Middle row: Results using the proposed method. A posteriori foreground PDF map obtained by the color module ($\mathcal{PF}_3$).

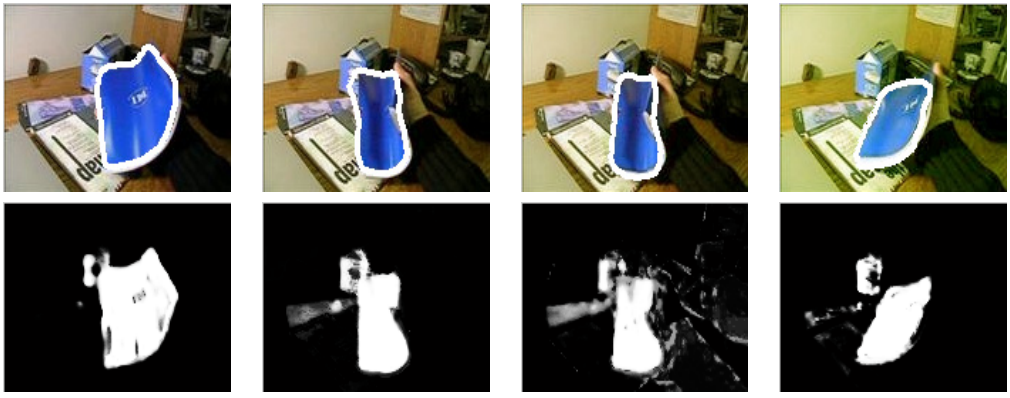
In the second experiment (Fig. 5.18) we show how our method performs in a real video sequence of an octopus changing its appearance while camouflaging. Observe that the foreground a posteriori probability maps of the color module give a rough estimate about the target position, especially when the octopus appearance is quite similar to the background. Nevertheless, a detailed detection of the target may be obtained by correcting the color estimate using the shape module.

In order to emphasize the importance of adapting simultaneously color and contour features using particle filters, in the rest of the experiments, the performance of the discussed algorithm will be compared to a tracking technique that uses multiple hypotheses to predict the contour of the object and accommodates the color with a predictive filter based on a simple smooth dynamic model such as:

$$\mathbf{g}^t = (1-\beta)\,\mathbf{g}^{t-2} + \beta\mathbf{g}^{t-1} \tag{5.30}$$

where $\mathbf{g}$ is the parameterization of the color distribution (with the same structure as in Eq. 5.20) and β is a mixing factor. Actually, this approach is quite similar to the ICondensation technique described in (52).

Tracking results using a smooth color dynamic model



Tracking results using the proposed method

Figure 5.19: **Experiment 3: Tracking results of a bending book in a sequence with smooth change of illumination**. Top row: Results using only a contour particle filter and assuming smooth change of color. The method fails. Middle row: Results using the proposed method. Bottom row: A posteriori object probability map of the color module ($\mathcal{PF}_3$).

Experiment 3 corresponds to the tracking of the non-rigid boundary of a bending book in a video sequence, where the lighting conditions smoothly change from natural lighting to yellow lighting. Fig. 5.19 shows some frames of the tracking results. Note that despite the smooth change of illuminant, the smooth dynamic model is unable to track the contour of the object. The reason of the failure is that the smooth dynamic model cannot cope with the effect of self-shadowing produced during the movement of the book.

5.6.2 Tracking under abrupt lighting changes

In Experiment 4, the color distribution of the bending book sequence previously presented, is manually modified in order to simulate an abrupt change of illumination. The top row of Fig. 5.20 shows three consecutive frames presented to the algorithm. Note the abrupt illumination change occurred between frames $t-1$ and t. Results prove the inability of the smooth color model to predict such a change, since the a posterior probability map of the foreground

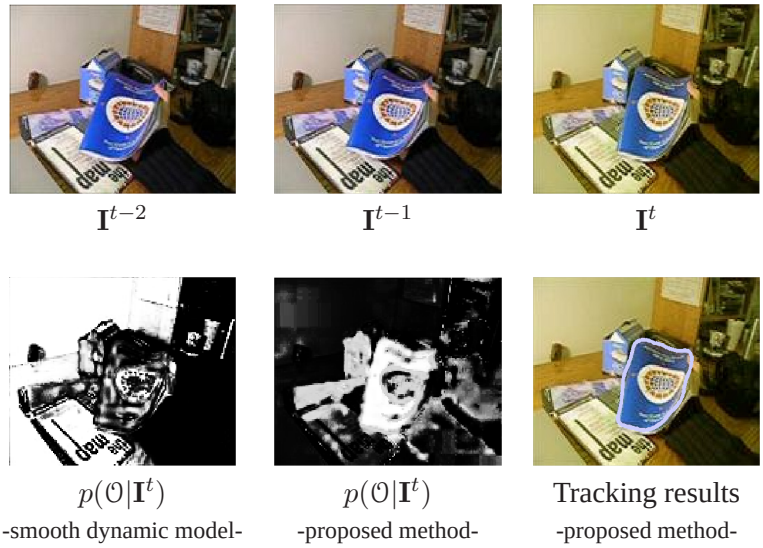


Figure 5.20: **Experiment 4: Tracking results of a non-rigid object (a bending book) in a sequence with abrupt changes of illumination.** Top row: $\mathbf{I}^{t-2}$, $\mathbf{I}^{t-1}$ and $\mathbf{I}^t$ are three consecutive images. Note the abrupt change in illuminant between frames $t-1$ and t. Bottom left: $p(\mathcal{O}|\mathbf{I}^t)$ map obtained assuming a smooth dynamic model of the color feature. There is no good discrimination between the foreground and background. Bottom center: $p(\mathcal{O}|\mathbf{I}^t)$ map provided by the proposed framework. The fore/background discrimination is clearly enhanced with respect to the smooth dynamic model case. Bottom right: Tracking results obtained after using $p(\mathcal{O}|\mathbf{I}^t)$ to eliminate false edges from image and fitting a deformable contour to the object boundary.

region depicted in Fig. 5.20 (bottom-left) does not discriminate between foreground and background, whereas a good result is obtained with the method proposed is this chapter (Fig 5.20, bottom-center and bottom-right).

In Fig. 5.21 (Experiment 5) similar results are presented when tracking a rigid object (the can) in a sequence that also suffers from abrupt illumination changes. Note again, that the smooth dynamic model to predict the color change is not appropriate.

In the final experiment (Experiment 6) we have tested the algorithm with the sequence of a moving leaf used as example throughout the whole chapter. Although this is a challenging sequence because it is highly cluttered, the illumination changes abruptly and the target moves unpredictably, we can perform the tracking with the proposed method. Fig. 5.22 shows some frames of the tracking results. Observe the abrupt change of illumination between the first and second frames, which leads to failure when we try to track using a contour particle filter with a smooth color prediction.

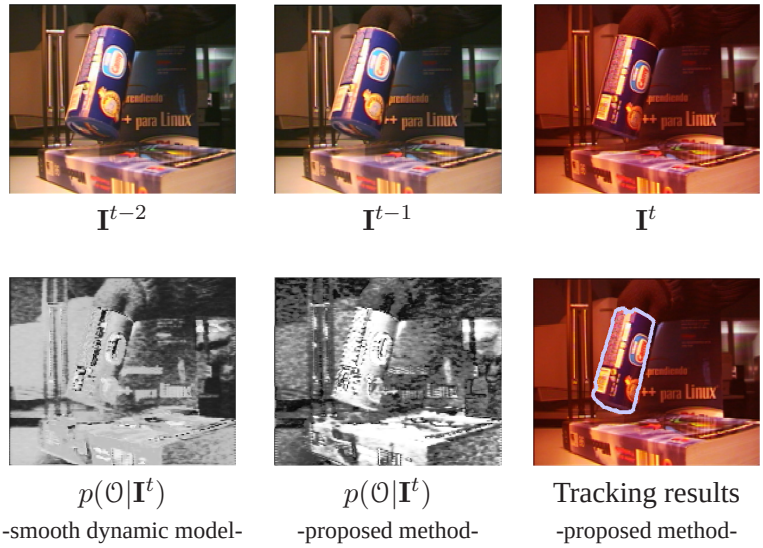


Figure 5.21: **Experiment 5: Tracking results of a rigid object (the can) in a sequence with abrupt changes of illumination.** See Fig. 5.20 for interpretation of results.

5.7 Summary

The use of various cues to represent the object permits the enhancement of visual tracking algorithms, and make them more robust to several artifacts existing in video sequences dealing with real and unconstrained environments. Nevertheless, most of the algorithms in the computer vision literature addressing the tracking problem by the integration of different cues, are based on specific heuristics, or do not take complete profit from the dependence between object features.

In this chapter, a general probabilistic framework allowing to integrate any number of object features has been described. The state of the features may be estimated by any algorithm which outputs a PDF (for instance particle filters or Kalman filter), and the method allows to integrate both dependent and independent features.

The proposed framework is theoretically proven and validated in a tracking example with synthetic data, which has been used as a bechmark to compare the performance of our method with other well-known algorithms from the field. The best results in terms of accuracy and reliability are obtained by the method presented here. Furthermore, in the specific case that the integrated features are estimated by particle filters, the method does not suffer from the *curse of dimensionality* problem, which usually affects particle filter formulations, producing

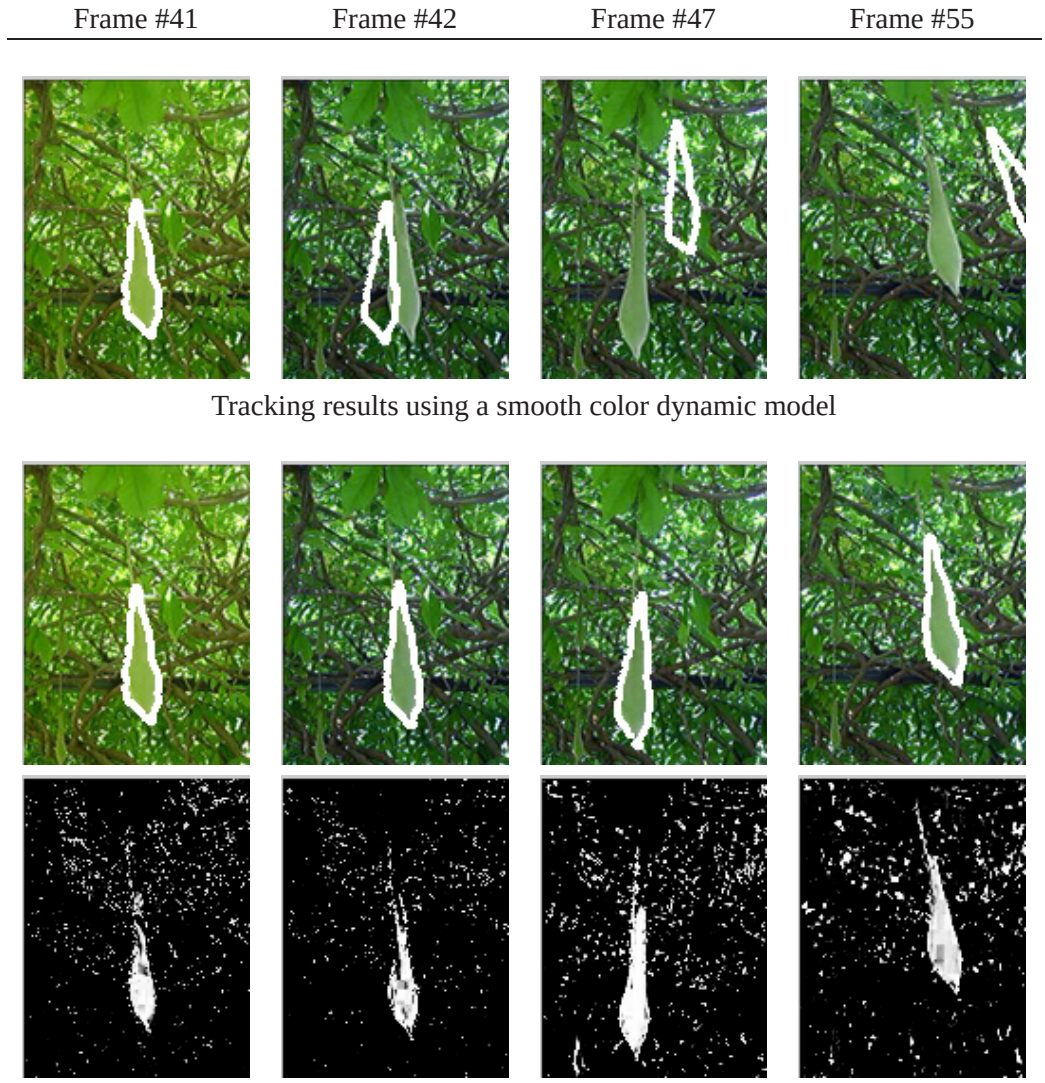


Figure 5.22: **Experiment 6: Tracking results of a leaf.** Tracking results of a cluttered sequence, where the target moves following unexpected paths. Furthermore, the sequence suffers from an abrupt change of illumination (observe it between Frame #41 and Frame #42). Top row: Results using a contour-based particle filter and assuming smooth change of the color feature. The method fails. Middle row: Successful results obtained using the method proposed in this chapter. Bottom row: A posteriori PDF map of the color module ($\mathcal{PF}_3$). Observe how the tracked leaf is clearly detected, and the unexpected illumination change does not destabilize the tracker.

exponential increases in the computation when the dimensionality of the state space increases.

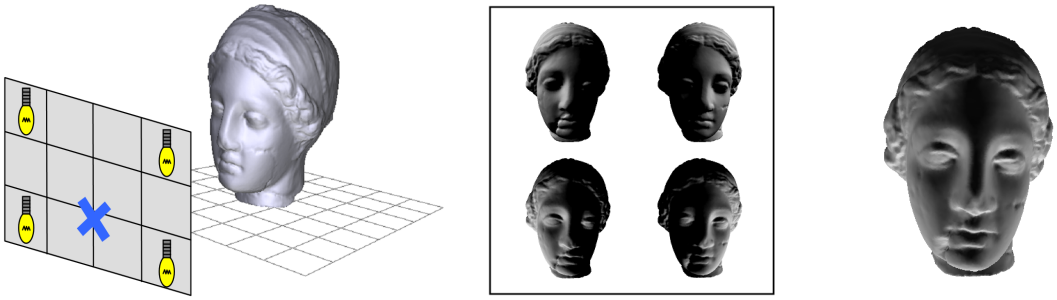
Finally, the features defined in Chapter 3 have been integrated in the framework in order to design a robust tracking algorithm that simultaneously accommodates the colorspace where the image points are represented, the color distributions of the object and background and the contour of the object. The effectiveness of the method has been proven by successfully tracking objects from synthetic and real sequences presenting high content of clutter, non-rigid boundaries, unexpected target movements and abrupt changes of illumination.

Chapter 6

Optimal illumination for video relighting

As we have mentioned in the introductory chapter, visual tracking is a tool utilized in a variety of tasks extending from autonomous vehicle navigation to video entertainment and virtual reality applications. This chapter will present an interesting application of the proposed tracking methodology concerning to video relighting. In the following sections, it will be explained how an 'image-based' approach may be used for relighting video sequences: images under new lighting conditions will be rendered from a linear combination of a set of pre-acquired images, illuminated by different (and known) light patterns. The main contribution of the chapter will be in the study of the optimal way to illuminate the scene, in the sense of determining which are the light patterns that need to be projected on the scenario, in order that using a minimum number of them, the best relighting results are obtained.

Within this framework, the tracking methodology suggested in previous chapter will play a fundamental role, since a very important part of the video relighting procedure refers to the pixel alignment in consecutive frames, where each frame corresponds to an image of the scene, illuminated by a distinct light pattern. These light patterns produce abrupt illumination changes between consecutive images. Chapter 5 has proved that the proposed tracking algorithm is appropriate to deal with such kind of difficulties. However, since the video relighting requires to solve the correspondence in consecutive images at a pixel level, an additional optical-flow stage will be necessary, although highly simplified after solving the first tracking stage. We will see that the combination of both techniques allows the alignment and subsequent relighting of the video sequences.



(a) (b) (c)

Figure 6.1: **Relighting a still object example.** Relighting is done by combining images acquired under known illumination. (a) The scene is illuminated by different light patterns, for instance single light sources (the bulbs). (b) Reference images acquired when the object is illuminated by the known light sources indicated in the plot (a). (c) A new image under unknown lighting conditions may be rendered by linear combination of the reference images. This example relights the object as if it were illuminated by a single light source placed on the blue cross of plot (a).

6.1 Introduction

Relighting images of still objects with unknown geometry has recently become a research topic of great interest in both computer vision and computer graphics (27; 28; 42; 68; 70; 97; 98; 106; 110; 138). The ability of photorealistically modeling the object, or part of the object, under changes in lighting may be used in many applications, such as object recognition and identification tasks or image-based rendering of objects and textures.

Most of the techniques use an image-based approach; the images under new lighting conditions are synthesized from a set of reference images previously acquired under known illumination conditions. Considering that the lighting process obeys the rules of superposition (20; 97), new images are generated via linear combination of the set of reference images (see Fig. 6.1).

A natural extension of the problem refers to the relighting of video sequences, which enables additional and interesting applications, mainly in the context of multimedia and entertainment. For instance, it could be applied to re-illuminate the action of an actor, recorded indoors, and illuminated with the light of an outdoor environment in a post-processing stage. However, this extension of the problem has only been covered by a few references, and existing results are constrained to highly controlled environments, where the moving object is clearly discriminated from the background and where the relighting method is not optimized. By re-

lighting optimization we mean to synthesize the images as close (in a L^2 sense) to real images as possible.

The relighting methodology used for re-illuminating video sequences is similar to the technique used for relighting static objects, i.e, a specific frame of the sequence under new lighting conditions is generated as a linear combination of reference images illuminated with known lighting conditions. Nevertheless, the movement of the object between consecutive reference frames introduces additional difficulties. First of all, consecutive images are geometrically warped, and corresponding points at different frames are misaligned. Secondly, since the orientation of the corresponding points at different frames with respect to the light sources is different, their appearance changes, even though the light position and intensity remains constant. In the following we will call this error, *orientation error*. All these difficulties make that the superposition principle cannot be directly applied in order to relight the frames of a video sequence. Previously, all of the reference images used to relight a specific frame, need to be aligned with respect to the same coordinate frame. Note that although a perfect alignment could be achieved, the orientation error cannot be removed. At the most we can reduce it by using a low number of reference images.

Therefore, the methodology for relighting video sequences involves two main subtasks: from one side, we need to seek for an alignment algorithm robust to abrupt illumination changes, since consecutive reference frames are illuminated by different lighting patterns, which produce abrupt changes on the temporal appearance of the object. On the other side, it is necessary to optimize the relighting process, such that the error in the rendering results is minimized using as few reference images as possible.

Next, we briefly describe each of this tasks, summarizing the main contributions of the chapter.

6.1.1 Optimal relighting of video sequences

It has been shown in the literature that image-based relighting of scenes with unknown geometry can be achieved through linear combinations of a set of pre-acquired reference images. Since the placement and brightness of the light sources can be controlled, it is natural to ask: what is the optimal way to illuminate the scene to reduce the number of reference images that are needed?

In this chapter we show that the best way to light the scene (i.e., the way that minimizes the number of reference images) is not using a sequence of single, compact light sources as is

most commonly done, but rather to use a sequence of lighting patterns as given by an *object-dependent* lighting basis. While this lighting basis, which we call the optimal lighting basis (OLB), depends on camera and scene properties, we show that it can be determined as a simple calibration procedure before acquisition. We demonstrate through experiments on real and synthetic data that the optimal lighting basis significantly reduces the number of reference images that are needed to achieve a desired level of accuracy in the relit images. This reduction in the number of needed images is particularly critical in the problem of relighting in video, as corresponding points on moving objects must be aligned from frame to frame during each cycle of the lighting basis.

We show, however, that the efficiencies gained by the optimal lighting basis simplify considerably the alignment problem, making the relighting in video possible. We present several relighting results on real video sequences of moving objects, moving faces, and scenes containing both. In each case, although a single video clip was captured, we are able to relight again and again, controlling the lighting direction, extent, and color.

6.1.2 Image alignment in dynamic environments

One of the main problems to solve when relighting moving objects refers to the alignment. Specifically, we need to align points across a window of frames so that when we superpose the reference frames in order to render an image under new lighting conditions, we do not blur the information from different points on the object. To do this, we estimate the optical flow over the sequence of images. This problem is made more difficult by the fact that the illumination varies from one frame to the next due to the use of a different lighting pattern for each frame. In recent works, usually the movement of the objects in dynamic and changing environments, is measured by means of structured light techniques (135; 147) or by using the *motion capture* technology, where some reflective or magnetic markers that can be easily tracked, are placed on the surface of the moving object (47). However, these kind of techniques are not valid for our purposes because they are invasive, in the sense that they might change the appearance of the object. In order to have a general and non-invasive application, we use passive computer vision techniques and the alignment is accomplished in two stages. First, using the methodology described in the previous chapter we apply a foreground/background segmentation over the moving regions, and second, each segmented region in consecutive frames, is aligned using a modification of the Lucas Kanade (73) optical flow algorithm, that increases the insensitivity of the algorithm to illumination changes.

After this short discussion, the rest of the chapter is organized as follows: In Section 6.2, we review related work on image and video relighting. In Section 6.3, we describe the relighting process of still and moving objects. In Section 6.4, we focus on the selection of the best illuminant basis for relighting. The performance of different light basis is analyzed for synthetic data in Section 6.5. The experiments are extended to real data in Section 6.6, where we show the relighting results of real video sequences. Precisely in this section we will discuss the proposed optical flow algorithm, which includes the tracking method described in the Chapter 5. Chapter summarization is given in Section 6.7.

6.2 Related work

A great deal of past work has focused on relighting scenes using pre-synthesized (31) or pre-acquired (27; 43; 46; 63; 78; 79) reference images. In each of these, the reference images are gathered by systematically varying the lighting direction. If the sampling of the lighting directions is dense enough, then due to the linearity of scene radiance, images of the scene under a user specified illumination can be synthesized by superposition of the single light source images, see again (31).

In nearly all of this work the reference images were acquired under single, compact source illumination. In (46), incandescent spot lights were used to sample 66 lighting directions on a sphere as reference images of a human face were gathered. In (43), xenon strobes were used to sample 64 lighting directions on a geodesic dome as reference images of a human face were gathered. In (27), a moving compact light source was used to sample 2048 lighting directions for the illumination of, yet again, a human face. Yet, here the density of the sampling allowed for impressively accurate results in the synthesis of effects such as specularities and cast shadows. In (79) compact light sources were used to sample 60 illumination directions per viewpoint for objects made of specular and fuzzy materials. And in (96), a moving spotlight was used to gather 4096 images of a still-life.

One of the aims of the work presented in this chapter is to show that illumination using single, compact light sources is not the most efficient for relighting. In contrast to much of this past work, we will show that if properly chosen lighting patterns are used to illuminate the scene, then many fewer reference images need be gathered. This is not the first work to consider using light patterns for relighting. However, all of the lighting bases that have been used in the existing literature to date are pre-chosen and are not a function of the camera or

scene properties. In (97) natural skylight illumination is approximated by a set of *steerable functions*. Schnechner et al. (110) use a scheme based on *Hadamard* codes for reducing Signal to Noise Ratio in the images. Debevec et al. (41) use terms of the *spherical harmonic* basis. An analysis of the efficiency of spherical harmonics for relighting can be found in (106); however, the optimality of a spherical harmonic basis holds only for objects with Lambertian reflectance and scenes without cast shadows.

Finally, this is not the first work to consider relighting in video. Debevec et al. (28) use a sphere of controlled light sources to light a moving object during acquisition with illumination pre-acquired from a different environment. While any illumination can be specified during acquisition, the resulting video sequence cannot be relit. In order to relight a moving face, (47; 98; 103) first fit 3-D models to the face shape and then used this to render frames under new illumination.

The recent work of Gardner et al. (41) is probably the closest to the video relighting component proposed in this chapter. Like ours, the goal of their work is to acquire a video sequence that can be relit again and again according to user specified illumination. To do this, (41) acquire and then process, as we do, a video sequence in which the lighting is systematically varied over the course of the sequence. Gardner et al. (41) use ten light patterns representing the first nine terms of spherical harmonics plus one directional light source to relight the face of an actor. In contrast, we develop and then use an object-dependent lighting basis that is significantly more efficient for video relighting. In our experiments, we have shown that we can reduce the number of light patterns that are needed by a factor of $2-3$. This is, as we will argue later, critical for the case of moving objects which require frame by frame alignment. A disadvantage of our method, however, is that it requires computation of the optimal lighting basis. Still, this can be accomplished within a few seconds prior to video capture.

6.3 Relighting with a lighting basis

In this section, we give a mathematical description of the relighting process using a lighting basis. We first define relighting for still objects and then introduce time dependence in the formulation in order to take into account the relighting of moving objects in video sequences.

6.3.1 Relighting in static scenes

Our setup is as follows. The scene is illuminated simultaneously by m single light sources, each of varying brightness; we call this illumination a light pattern. We assume that the light sources are distant, so they can be parameterized as a function of direction only. Let the m-dimensional array $\mathbf{l}_p = [L_p(\theta_1, \phi_1), \ldots, L_p(\theta_m, \phi_m)]^T$ be the vector of radiances of all the single light sources generating the p-th lighting pattern, where $L_p(\theta_l, \phi_l)$ is the radiance of the l-th light source of the p-th light pattern, and $\mathbf{\Phi}_l = [\theta_l, \phi_l]^T$ are the global spherical coordinates of the l-th light source.

In order to compute the image of a pixel $\mathbf{u}_i = [u_i, v_i]^T$ we make use of the properties of image superposition (20; 97; 137):

1. The image resulting from multiplying each pixel by a factor α is equivalent to an image resulting from a light source with intensity multiplied by the same factor.
2. An image of a scene illuminated by two light sources $L(\mathbf{\Phi}_1)$ and $L(\mathbf{\Phi}_2)$, equals the sum of an image illuminated with $L(\mathbf{\Phi}_1)$ and another image illuminated with $L(\mathbf{\Phi}_2)$.

From these properties, we compute the image of a pixel $\mathbf{u}_i$ under the light pattern $\mathbf{l}_p$ as follows

$$\mathbf{i}_p(\mathbf{u}_i) = \sum_{l=1}^{m} R_{\mathbf{u}_i}(\mathbf{\Phi}_l) L_p(\mathbf{\Phi}_l) = \mathbf{r}_{\mathbf{u}_i}^T \mathbf{l}_p \tag{6.1}$$

where $\mathbf{r}_{\mathbf{u}_i} = [R_{\mathbf{u}_i}(\mathbf{\Phi}_1), \ldots, R_{\mathbf{u}_i}(\mathbf{\Phi}_m)]^T$ is an m-dimensional vector with the elements $R_{\mathbf{u}_i}(\mathbf{\Phi}_l)$ being the *reflectance* of pixel $\mathbf{u}_i$ as a result of illumination from direction $\mathbf{\Phi}_l$, see again (27).

The above equation can be extended in order to consider all the n image pixels

$$\mathbf{i}_p = \mathbf{R}\mathbf{l}_p \tag{6.2}$$

where $\mathbf{i} \in \mathbb{R}_{n\times 1}$ contains all image points (gray level images arranged in a vector form)[1], and $\mathbf{R} = [\mathbf{r}_{\mathbf{u}_1}, \ldots, \mathbf{r}_{\mathbf{u}_n}]^T$ is an $n \times m$ matrix of reflectance functions for all image points (we call it *reflectance matrix*). Note that the rows of $\mathbf{R}$ denote image pixels, while the columns correspond to different light source positions.

Now the collection of p reference images of the scene illuminated by p lighting patterns can be expressed by

$$\mathbf{I}_\mathbf{L} = \mathbf{R}\mathbf{L} \tag{6.3}$$

[1] Observe that we consider gray level images, and therefore they can be represented as a $n \times 1$ vector '$\mathbf{i}$'.

where $\mathbf{I_L} = [\mathbf{i}_1, \ldots, \mathbf{i}_p]$ is an $n \times p$ matrix containing the images of the object under different lights and $\mathbf{L} = [\mathbf{l}_1, \ldots, \mathbf{l}_p]$ is an $m \times p$ matrix representing the different lighting patterns used to illuminate the object. We now need to determine how the image of the scene illuminated by lighting patterns can be decoded into images of the scene illuminated by single light sources.

Consider acquiring a set of m reference images each of which is illuminated by a single point light source. The lighting in the i-th image can be represented by a vector $\mathbf{e}_i$ in which the i-th element has the value 1 and the remaining elements have the value 0. The matrix of all m single light source patterns can be written as $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_m]$. Note that these light source patterns, as given by the columns of $\mathbf{E}$, form the standard basis; note also that $\mathbf{E}$ is the identity matrix. Now the images formed by these single light source patterns can be written as

$$\mathbf{I_E} = \mathbf{R}\,\mathbf{E} \tag{6.4}$$

where $\mathbf{R}$ is the reflectance matrix described earlier.

We need to find the linear transformation $\mathbf{D}$ that will decode the reference images acquired using the lighting patterns to recover the images that would be created under single light source illumination $\mathbf{E}$. By decode we mean we can find $\mathbf{I_E}$ as

$$\mathbf{I_L}\,\mathbf{D} \;=\; \mathbf{R}\,\mathbf{L}\,\mathbf{D} \;=\; \mathbf{R}\,\mathbf{E} \;=\; \mathbf{I_E} \tag{6.5}$$

If the lighting patterns are linearly independent and the number of patterns p is greater than or equal to the number of single light sources m, then we can decode the lighting patterns exactly using the decoder matrix $\mathbf{D} = \mathbf{L}^{-1}$. If the number of lighting patterns p is less than the number of single light sources m then the $rank(\mathbf{L}) < m$. In this case we cannot invert the matrix of lighting patterns $\mathbf{L}$ and must settle for an approximate decoding as given by $\mathbf{D} = \mathbf{L}^{+} = (\mathbf{L}^T\mathbf{L})^{-1}\mathbf{L}^T$. In both cases, we write

$$\mathbf{I_D} \;=\; \mathbf{I_L}\,\mathbf{D} \;\approx\; \mathbf{RE} \tag{6.6}$$

where $\mathbf{I_D}$ is the matrix of decoded reference images.

Relighting can then be achieved taking the desired linear combinations of the decoded images $\mathbf{I_D}$. For example, imagine you want to relight a scene with user specified illumination $\mathbf{l}_{new}$, we get the image of the scene under this illumination as given by $\mathbf{i}_{new} \;=\; \mathbf{I_D}\mathbf{l}_{new} \;\approx\; \mathbf{R}\mathbf{l}_{new}$.

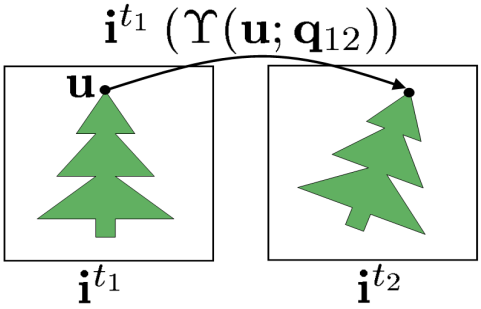


Figure 6.2: **Alignment of images $\mathbf{i}^{t_1}$ and $\mathbf{i}^{t_2}$.**

6.3.2 Relighting in video

For relighting moving objects, we illuminate the object with a sequence of p lighting patterns, synchronizing the lighting system with the camera, such that each image of the object is acquired with a single light pattern. To relight the video in a post-processing stage we first need to perform an optical flow alignment between consecutive frames. However, at this point we will delay the details of the optical flow alignment method until later in the chapter.

Let $\mathbf{i}^{t_1}(\mathbf{u})$ denote a frame acquired at time t_1 under illumination given by lighting pattern $\mathbf{l}_{\mathrm{mod}\ (t_1,p)}$ where $\mathrm{mod}\ (t_1,p)$ is remainder of t_1 divided by p. This $\mathrm{mod}\ (\cdot)$ addresses the fact that we are cycling through the p patterns over the course of the sequence. Let $\mathbf{i}^{t_1}(\Upsilon(\mathbf{u};\mathbf{q}_{12}))$ denote the frame $\mathbf{i}^{t_1}(\mathbf{u})$ acquired at time t_1 but warped in such a way that it is aligned with $\mathbf{i}^{t_2}(\mathbf{u})$ acquired at time t_2. The warping function $\Upsilon(\mathbf{u};\mathbf{q}_{12})$ takes the pixel $\mathbf{u}$ in the time frame of $\mathbf{i}^{t_1}$ and maps it to the subpixel location $\Upsilon(\mathbf{u};\mathbf{q}_{12})$ in frame $\mathbf{i}^{t_2}$. Note that $\mathbf{q}_{12}$ is the vector of warping parameters needed for the mapping from t_1 to t_2 (see Fig. 6.2 for an interpretation of the warping function).

In order to decode the lighting patterns at any given time t, we require a set of frames of the scene – in the pose of frame t – taken under p different lighting patterns. To do this, we take a window of p frames centered at frame t and align each of these $p-1$ frames to frame t. (We align only $p-1$ as frame t is already aligned.) Let the first frame in the window be called frame t_1, let the middle frame be t, and let the last frame be t_p. This gives a matrix of p aligned frames that can be written as follows:

$$\mathbf{I}_{\mathbf{L}}^{t} = \left[\mathbf{i}^{t_1}(\Upsilon(\mathbf{u};\mathbf{q}_{1t})),\ldots,\mathbf{i}^{t_p}(\Upsilon(\mathbf{u};\mathbf{q}_{pt}))\right],$$

where $t_i = t - \lfloor \frac{p}{2} \rfloor + i - 1$. Now to relight the video sequence, we decode each frame using

the same decoding matrix $\mathbf{D}$ to get $\mathbf{I}_{\mathbf{D}}^{t} = \mathbf{I}_{\mathbf{L}}^{t}\mathbf{D}$. Finally every frame can be relit much like the static case. If the user specifies the lighting at time t as $\mathbf{l}_{new}^{t}$, we can compute the relit image in frame t as $\mathbf{i}_{new}^{t} = \mathbf{I}_{\mathbf{D}}^{t}\mathbf{l}_{new}^{t}$.

6.3.3 Sources of error

If the object being relit remains static, error in relighting and decoding arises only if we use a subset of the lighting basis, i.e., we use p m-dimensional vectors (where $p < m$) to span the m-dimensional space of lights. This error, which we call the *sub-basis error*, is reduced by using a higher number of light patterns and converges to zero when $p = m$.

If the object moves, we need to consider two additional sources of error. There is intrinsic error in the alignment from the optical flow algorithm; we call this the *alignment error*. Since we need only to align the images that are inside a temporal window of length p, this error is reduced by decreasing the size of the temporal window, i.e., by decreasing the number of lighting patterns. This error can also be decreased by increasing the frame rate of the camera. In our experiments we used a camera capable of acquiring 30 frames per second (fps), but faster cameras are readily available.

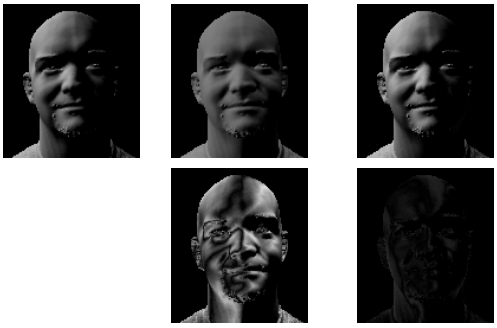
Finally, there is the error produced as an object in the scene changes its orientation with respect to the camera or light sources; we call this the *orientation error*. Even if the displacement in the scene is perfectly realigned with the alignment algorithm, the displacement itself can induce relative orientation change of surface points with respect to the camera and the light sources. A simple rotation of the object between frames will induce this error. And, if the camera and light source are close to the scene, a translation of the object can induce this error as well. As with the alignment error, the orientation error can again be reduced either by using a low number of lighting patterns p or by increasing the camera's frame rate.

Fig. 6.3 depicts all three of these sources of error for an experiment using images generated from a synthetic 3-D head. As the data used was synthetic, we were able to isolate and separately display each source of error. A brief discussion of each is included within the figure.

6.4 Selecting the optimal lighting basis

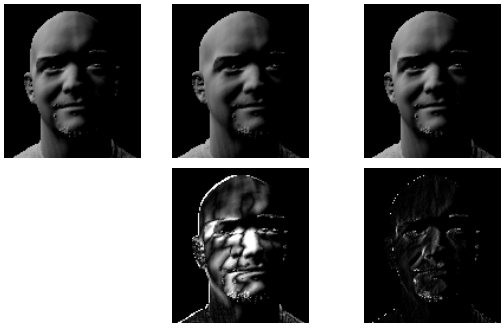
We now concentrate on the selection of the optimal lighting basis $\mathbf{L}$ for relighting video sequences. As we have shown at the end of the previous section, the alignment and orientation errors can be reduced with the use of fewer lighting patterns. However, the sub-basis error

Sub-Basis Error



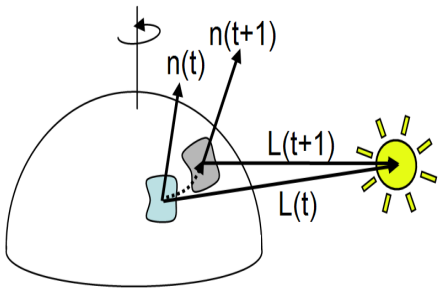
Sub-Basis Error: Error produced when using a reduced number of lighting patterns. Upper row, left to right: Ground truth image; image decoded using only 3 lighting patterns; image decoded using 9 lighting patterns. Lower row: Error when using 3 lighting patterns and 9 lighting patterns, respectively.

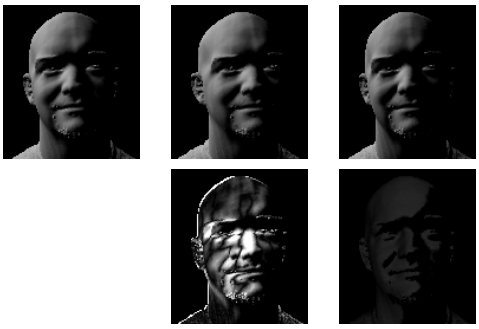
Alignment Error



Alignment Error: Error produced by inaccuracies in the optical flow alignment. Upper row: $\mathbf{i}^{t_1}$; $\mathbf{i}^{t_2}$; aligned image $\mathbf{i}^{t_2}(\Upsilon(\mathbf{u};\mathbf{q}_{21}))$. Lower row: Error before the alignment $|\mathbf{i}^{t_1} - \mathbf{i}^{t_2}|$; Error after the alignment, $|\mathbf{i}^{t_1} - \mathbf{i}^{t_2}(\Upsilon(\mathbf{u};\mathbf{q}_{21}))|$. This error will increase as the displacement in the scene becomes larger; thus it will increase as more lighting patterns are used.

Orientation Error





Orientation Error: The displacement of a patch between frame t_1 and t_2 can cause a change in orientation with respect to the camera or light sources. This will cause errors as described below even when perfect geometric alignment is achieved before decoding the image. **Left:** Diagram showing how the movement between times t_1 and t_2 of a patch w.r.t. a light source effects its appearance. When frames t_1 and t_2 are used to decode image values at point $\mathbf{u}$ they induce error. **Right, upper row:** $\mathbf{i}^{t_1}$; $\mathbf{i}^{t_2}$; aligned image $\mathbf{i}^{t_2}(\Upsilon(\mathbf{u};\mathbf{q}_{21}))$. In this case, we assume a *perfect* alignment, so that the error is produced only by the change of appearance of the head because of its relative movement w.r.t. the light sources. **Right, lower row:** Error before alignment, $|\mathbf{i}^{t_1} - \mathbf{i}^{t_2}|$; error even after perfect alignment, $|\mathbf{i}^{t_1} - \mathbf{i}^{t_2}(\Upsilon(\mathbf{u};\mathbf{q}_{21}))|$. This error will also increase as the displacement in the scene becomes larger; thus it will increase as more lighting patterns are used.

Figure 6.3: **Sources of error when relighting video sequences.**

increases as the number of lighting patterns decreases. Therefore, we need to find the lighting basis that minimizes the sub-basis error.

Our goal is to synthesize images of the scene under new illumination conditions such that the synthesized images are as close (in an L^2 sense) to real images as possible. Equivalently, for a desired level of accuracy, we want to find the lighting basis that minimizes the number of reference images that need to be acquired – for reasons detailed in Section 3.3. It is important to note that this optimal lighting basis is a complex function of camera and scene properties and, thus, is what we call an *object-dependent* lighting basis. Yet, we will show subsequently that this optimal basis can be determined using singular value decomposition (SVD) on images gathered during a calibration step before acquisition. For a typical scene this calibration can be done in a matter of seconds before video capture.

Consider again acquiring a set of m reference images each of which is illuminated by a single, compact point light source. The lighting in the i-th image can be represented by a vector $\mathbf{e}_i$ in which the i-th element has the value 1 and the remaining elements have the value 0. The matrix of all m single light source patterns can be written as $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_m]$. Note also $\mathbf{E}$ is the identity matrix.

Now the images formed by these single light source patterns can be written as

$$\mathbf{I_E} = \mathbf{R}\,\mathbf{E} \tag{6.7}$$

where $\mathbf{R}$ is the reflectance matrix described earlier.

Now let's say that instead of illuminating the scene with a sequence of single light sources as given by $\mathbf{E}$, we illuminate the scene with the optimal lighting basis denoted by $\mathbf{L}^*$. Under this illumination we get a different set of reference images $\mathbf{I_{L^*}}$ as

$$\mathbf{I_{L^*}} = \mathbf{R}\,\mathbf{L}^* \tag{6.8}$$

Now there exits a linear transformation $\mathbf{D}^* = (\mathbf{L}^*)^{-1}$ that will decode the reference images acquired using the optimal lighting basis to recover the images that would be created under single light source illumination $\mathbf{E}$. By decode we mean we can find $\mathbf{I_E}$ as

$$\mathbf{I}^*_\mathbf{L}\,\mathbf{D}^* \;=\; \mathbf{R}\,\mathbf{L}^*\,\mathbf{D}^* \;=\; \mathbf{R} \;=\; \mathbf{I_E} \tag{6.9}$$

But how is the optimal basis chosen given that we have measured $\mathbf{I_E}$?

Recall that we want to find the lighting basis that minimizes the number of reference images that need to be acquired. Our goal is to acquire many fewer than m reference images, yet

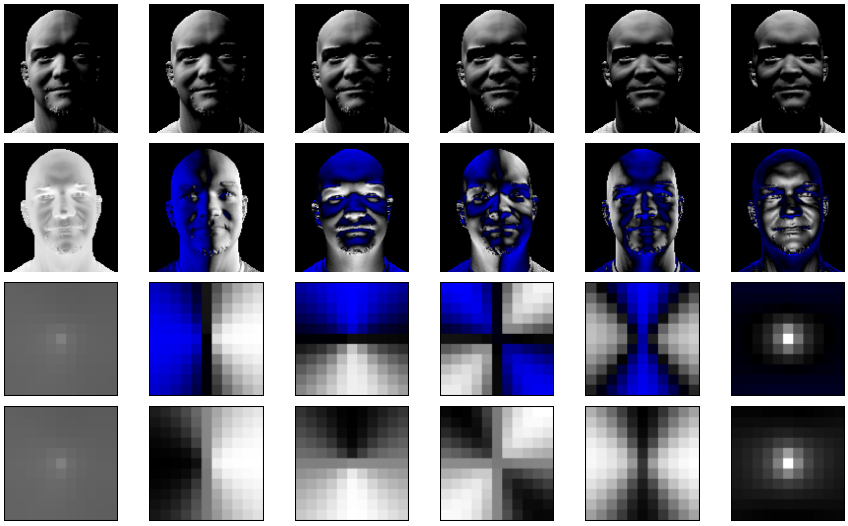


Figure 6.4: **Computing the optimal lighting basis using SVD.** First row: Images of the object illuminated by a single light source in different positions (columns of matrix $\mathbf{I_E}$). Second row: Optimal image basis (columns of matrix $\mathbf{U}$). They contain both positive values, shown in grey, and negative values, shown in blue. Third row: Lighting patterns from the optimal lighting basis (rows of matrix $\mathbf{L}^*$). They also contain positive and negative values. Observe the correspondence between the selected optimal lighting basis and the images basis shown in the upper row. Fourth row: Offset and scaling of the optimal lighting basis in order to make all its values positive.

still be able to decode these images to approximate, with the highest possible accuracy, the full set of reference images under point source illumination. If we perform a singular value decomposition (SVD), we can write

$$\mathbf{I_E} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{6.10}$$

where $\mathbf{U}$ is an orthogonal matrix; $\mathbf{S}$ is a diagonal matrix whose non-zero elements are singular values in decreasing order of $\mathbf{I_E}$; $\mathbf{V}^T$ is an orthogonal matrix. We choose as the optimal basis $\mathbf{L}^* = \mathbf{V}$ and the optimal decoding matrix $\mathbf{D}^* = \mathbf{V}^T$; we claim they are optimal in the following sense.

Let $\mathbf{L}$ be an $m \times p$ matrix formed from the first p columns of $\mathbf{L}^*$. Likewise let $\mathbf{D}$ be a $p \times m$ matrix formed from the first p rows of $\mathbf{D}^*$. Finally, let the p images acquired under illumination $\mathbf{L}$ be denoted again as $\mathbf{I_L}$. We then write the following approximation

$$\mathbf{I_E} \cong \mathbf{I_D} = \mathbf{I_L}\,\mathbf{D} \tag{6.11}$$

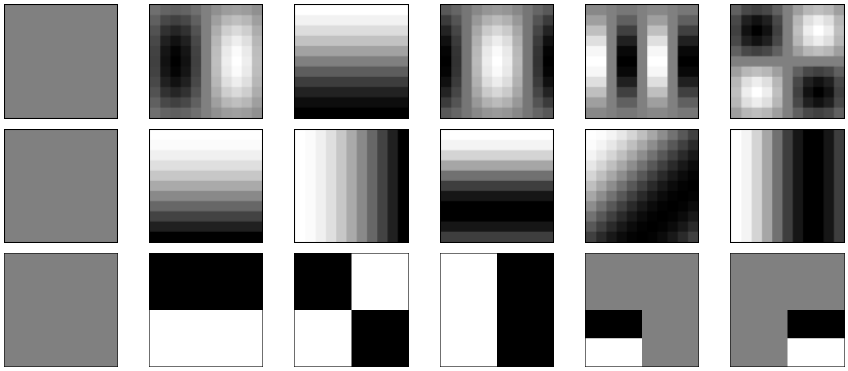


Figure 6.5: **The first 6 patterns of three object-independent lighting bases.** Spherical harmonics (first row), Fourier (second row) and Haar basis (third row). Compare these bases to the optimal lighting basis in Fig.6.4.

Now for any choice of p, the lighting patterns $\mathbf{L}$ extracted from the optimal lighting basis $\mathbf{L}^*$ minimize the sub-basis error in the above approximation, i.e., minimize the Frobenius norm $\|\mathbf{I_E} - \mathbf{I_D}\|$.

Note that since the matrix $\mathbf{L}^*$ contains negative values in some elements, we must offset and scale each basis to range between 0 and 1 to make a physically feasible lighting basis. Fig. 6.4 shows an optimal lighting basis for a synthetic 3-D head computed from a superset of the images in the first row. Compare this optimal lighting basis to the bases shown in Fig. 6.5.

6.5 Experiments with synthetic data

We now show, using several experiments with synthetic data, that the scene-dependent optimal lighting basis (OLB) performs better than the Fourier lighting basis (FLB), Haar lighting basis (HaLB) and spherical harmonic lighting basis (SHLB). We present these results for both static as well as moving objects. In the case of moving objects, since we are using synthetic data, we can assume perfect alignment so as to focus on the sub-basis errors produced by the different types of bases.

6.5.1 Performance comparison for static objects

In the first experiment, we compare the four lighting bases: FLB, HaLB, SHLB, and OLB. We use the bases as illumination patterns to render images of static synthetic objects. Then, for each case, we recover (decode) single light source images from the rendered images. These

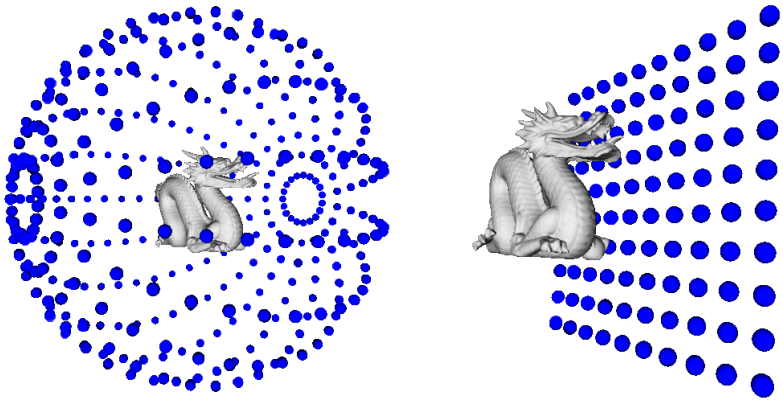


Figure 6.6: **The two different configurations of the light sources used in the synthetic experiments.** Left: Light sources lying on a sphere. Used for the comparison of the OLB with SHLB lighting patterns. Right: Light sources lying on a plane. Used for the comparison of the OLB with the Fourier and Haar lighting patterns.

recovered images are compared with rendered single light source images (ground truth) to compare the performances of the lighting bases. This experiment is done for several objects: a glossy sphere, and three non-convex objects: a human face, a buddha's statue and a dragon (courtesy of Cyberware). These models are assumed to have Lambertian reflectance with constant albedo.

When comparing the optimal bases with the Fourier and Haar bases, we assume that there are 12×12 light sources that lie on a plane. The object is assumed to be placed in front of the plane. On the other hand, since spherical harmonics are suitable only for use on a sphere, to make our comparison fair, both OLB and SHLB patterns are represented through 20×20 light sources lying on a whole sphere (see Fig. 6.6).

Our comparison is done using the following steps for each of the lighting bases (FLB, HaLB, SHLB, and OLB):

1. Render images of the object using single light sources.
2. Render images of the object using the lighting basis.
3. Decode single light source images of the object from the lighting basis images, as explained in Section 6.3.
4. Compute the sub-basis error (using the Frobenius norm) between the decoded and ground truth single light source images.

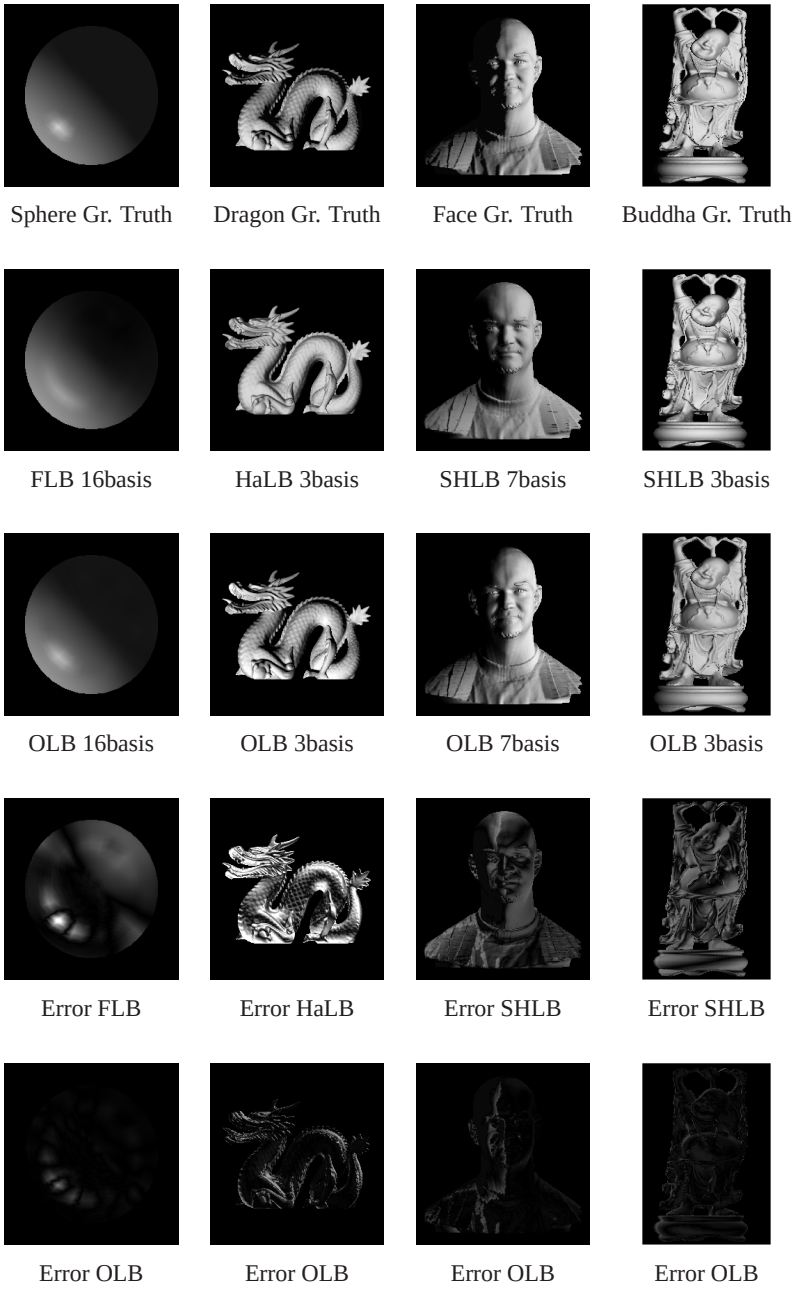


Figure 6.7: **Examples of reconstructed images, and reconstruction error for the synthetic static experiments.** Each column corresponds to a different experiment. For these examples we see that with the same number of basis images, the optimal lighting basis performs much better than the Fourier, Haar, and spherical harmonics lighting basis.

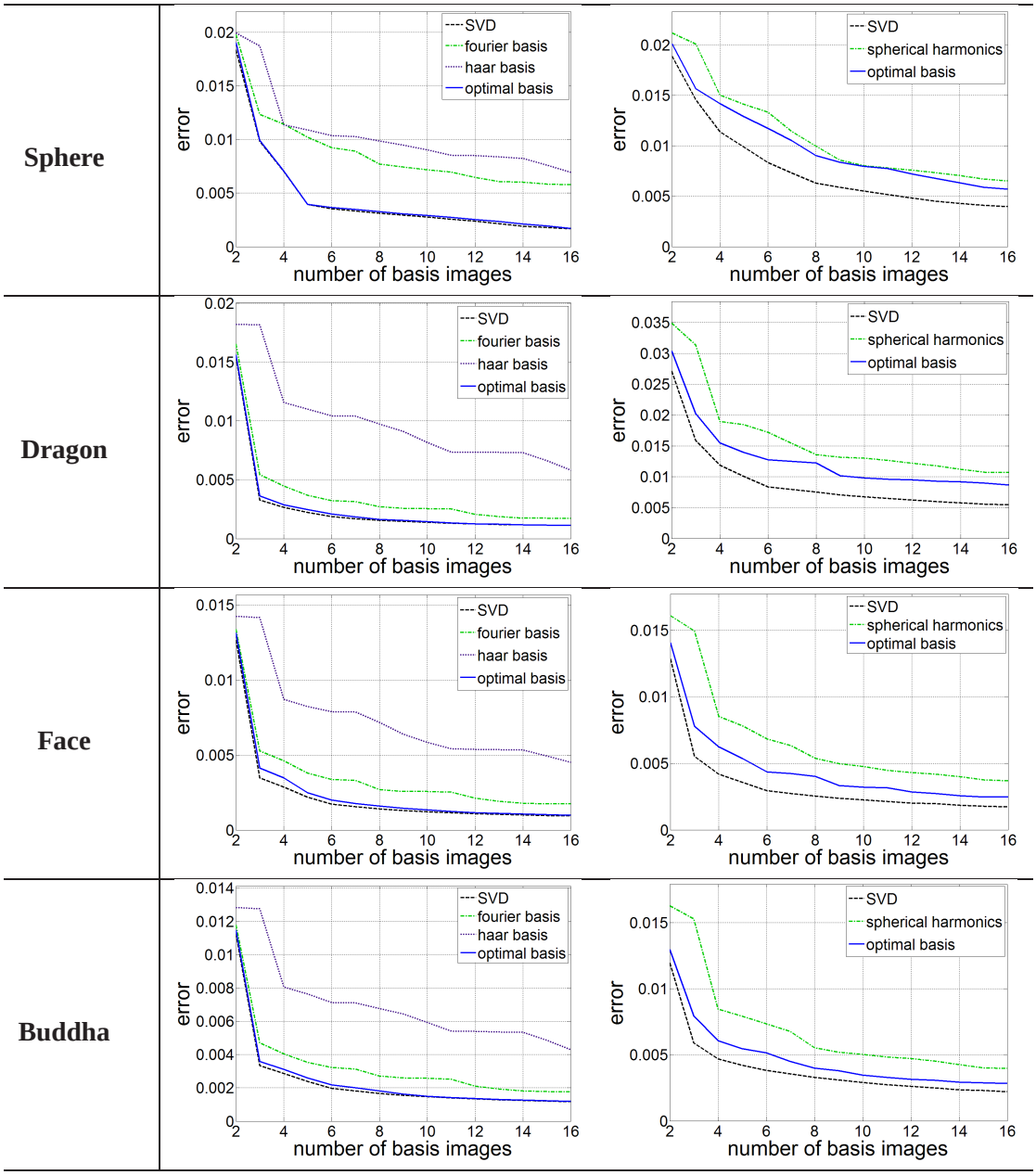


Figure 6.8: **The sub-basis errors in the synthetic experiments** for the different types of lighting bases plotted as a function of the number of basis images, for the synthetic sphere, dragon, face and buddha statue. Here, we have included plots for SVD approximation of the original data (see text for details).

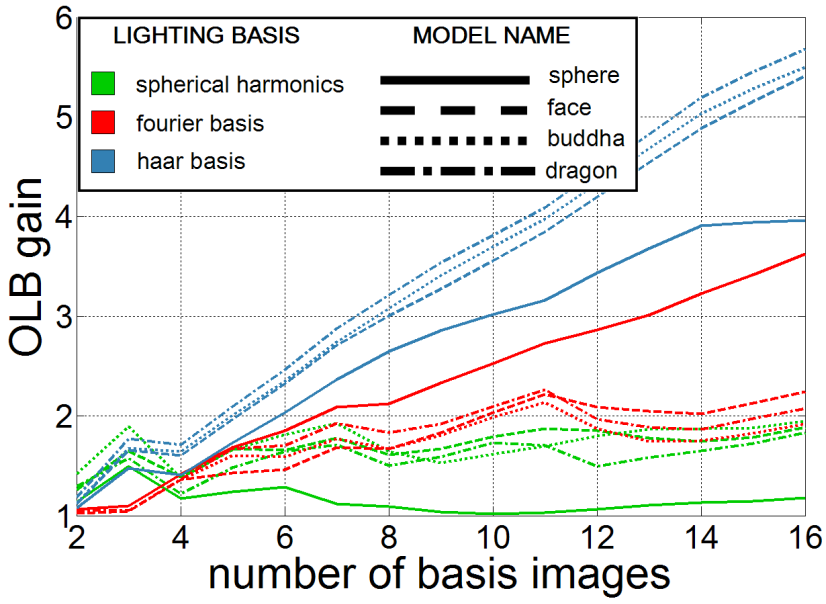
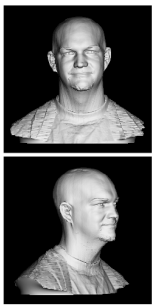


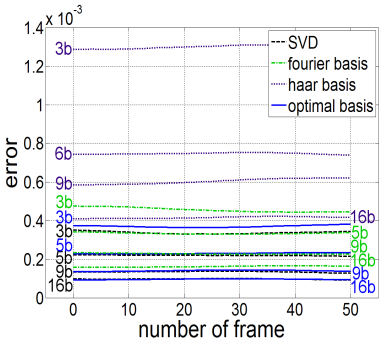
Figure 6.9: **Gains of the OLB with respect to all the other lighting basis**, (for all synthetic and static experiments), plotted as a function of the number of basis images used. For any given number of OLB images, the corresponding number of images of any other lighting basis that are needed to achieve the same reconstruction error equals the gain value. For instance, in the 'buddha' experiment instead of 6 optimal basis images, we will need to use $6 \times 1.8 \approx 11$ SHLB images, $6 \times 1.5 \approx 9$ FLB images or $6 \times 2.3 \approx 14$ HaLB images. Note that we cannot directly compare the results of the SHLB with the results of the FLB (or HaLB), because the former have been obtained with the lights lying on a sphere, while in the later, the lights lie on a plane.

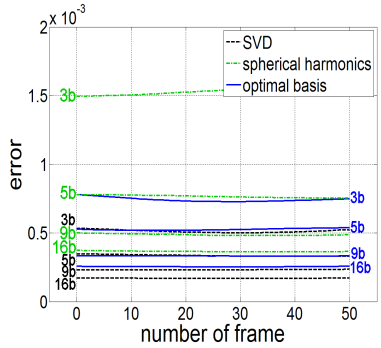
Fig. 6.7 shows several examples where the differences between the decoding results obtained using the optimal lighting basis and the spherical harmonics, Haar or Fourier lighting basis are clearly visible, especially when the object has a 'complex' geometry.

Fig. 6.8 shows the sub-basis errors for the different types of lighting bases in each of the experiments, plotted as a function of the number of basis images. We have also included the errors obtained when approximating the ground truth images using the most significative eigenvectors, resulting from the SVD decomposition of the single light source images. Note that these eigenvector images are not physically feasible through a relighting process because they contain negative values. We include these results just as a baseline case for comparison.

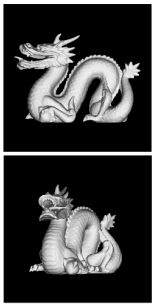
Fig. 6.9 shows the gains of the OLB with respect to all the other lighting basis plotted as a function of the number of basis images used. For any given number of optimal lighting basis

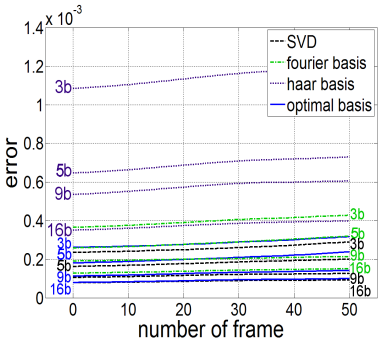


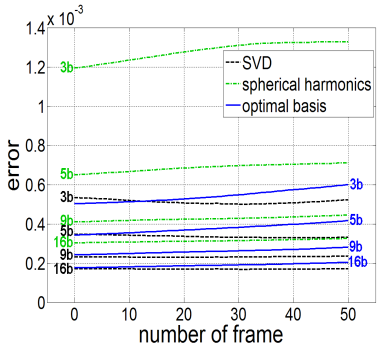




First and last frames — Sequence of the rotating face







First and last frames — Sequence of the rotating dragon

Figure 6.10: **Experiment with synthetic moving objects for the 'male' and 'dragon' experiments**. Left: the first and last images in the rendered sequences. Right: Reconstruction error for both experiments, plotted as a function of the frames of the sequence (horizontal axis), for the different lighting basis types and different numbers of basis images used for each type (the number next to each plot).

images, the corresponding number of images of any other lighting basis that are needed to achieve the same reconstruction error equals the gain value. It is clear from this plot that the optimal lighting basis is significantly more efficient than the others.

6.5.2 Performance comparison for moving objects

The second experiment relates to decoding rendered video sequences of the 'face' and 'dragon' models, that rotate around the vertical axis. Again, we assume that the object surfaces have Lambertian reflectance with constant albedo. The rendered sequences have 50 frames, and the rotation between consecutive frames is 1 degree. Just like in the experiment with static objects, when comparing the performance of the SHLB and OLB, the object is assumed to be lit by

a spherical source, while when comparing the OLB with the HaLB and FLB, the sources are placed on a plane. The illumination pattern is varied from one frame to the next based on the chosen lighting basis.

In Fig. 6.10(left) we show the first and the last frames of the sequences. Since these are synthetic examples, the alignment between frames is known to us and the only sources of error in the decoded images is due to the use of a lower number of basis images (sub-basis error) and the change in surface orientation with respect to the camera and the light sources (orientation error). Fig. 6.10(right) shows the reconstruction error (computed as an average over reconstructions of all the single sources) plotted as a function of the frames of the sequence, for each of the experiments. Note that the plotted error includes the sub-basis and the orientation errors. In each experiment, the different plots correspond to different lighting bases types and different numbers of basis images used for each type.

From the plots in Fig. 6.10(right), we can see that optimal lighting basis performs much better than SHLB, FLB and HaLB, even in video sequences. For instance, in the synthetic face example, with 5 images of the optimal lighting basis we obtain decoding results similar to using 9 spherical harmonic, 9 Fourier and 16 Haar basis images. Note that the optimal lighting basis was computed using rendered single source images of only the initial orientation (seen in the first frames) of the face and dragon. Even though these bases were used for all other orientations, we see that the reconstruction error does not increase noticeably as the objects rotate.

This efficiency of our optimal lighting basis is critical in the context of video relighting. The smaller the number of required lighting patterns, the easier it is to align the images and the lower is the orientation error. In other words, a lighting basis with lower sub-basis error naturally results in lower alignment and orientation errors.

6.6 Experiments with real scenes

In the preceding section, we have shown that our lighting basis is optimal for relighting moving objects in an ideal setting: Lambertian objects having constant albedo, linear light sources with equal power, a linear image acquisition system and no errors in the alignment. We now report experiments with real scenes. We use a setup that is calibrated to satisfy many of the assumptions we have made. We then apply our relighting method to static and dynamic scenes

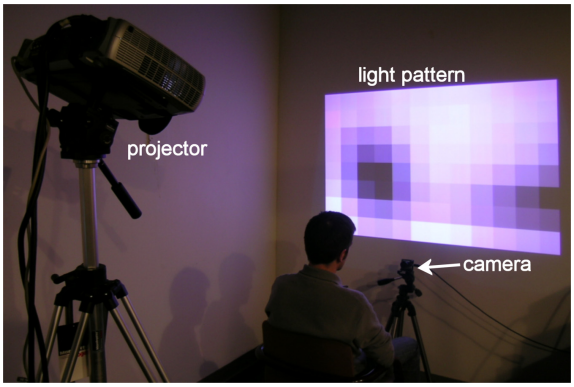


Figure 6.11: **Experimental setup used for the real experiments.**

that include non-Lambertian surfaces. In each case we show that the use of the optimal lighting basis enables us to produce relit videos of high quality.

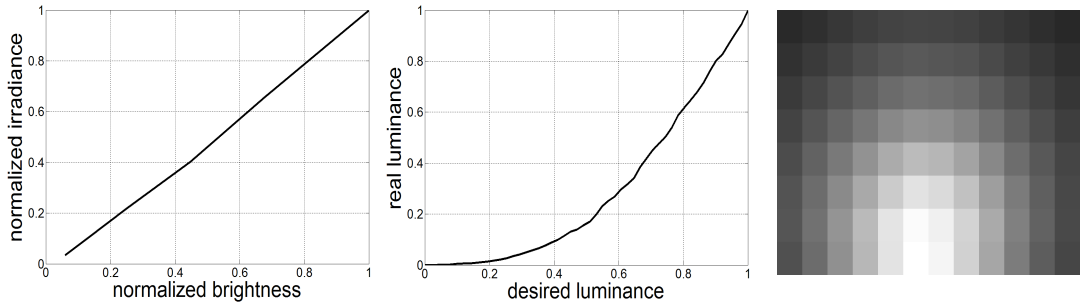
6.6.1 Experimental setup

Our setup is based on the system described in (110). The components of our setup are a color camera (Dragonfly IEEE-1394, color, 640×480 pixels) running at 30fps and a PC-controlled projector (Infocus LP820). The projector projects the basis patterns on a white wall, which in turn illuminates the scene (see Fig. 6.11). The scene is captured using the camera which is synchronized with the projector. The complete system has frame rate of 22fps. Note that a significantly higher frame rate can be achieved using a high-speed camera and a projector with a higher refresh rate. However, even with the current system, the efficiency of the optimal lighting basis allows us to capture and relight scenes with objects that move at reasonable speeds.

Since our setup uses a planar surface as the source area, we will only compare the results of using the optimal lighting basis with the Fourier and Haar lighting basis. Note that spherical harmonics are inappropriate for such a setup as they are defined over the sphere.

6.6.2 System calibration

One of the key assumptions we have made is that the light sources are linear and that the camera has a linear response. To this end, we have measured the radiometric response functions of the projector and the camera and used these response functions to linearize our system. Fig. 6.12a and Fig. 6.12b plot the response of these devices. We can verify that the response of the



(a) (b) (c)

Figure 6.12: **Camera and projector calibration.** A basic assumption we have made in our experiments is linearity. Therefore, the camera and projector response needs to be calibrated such as the linearity is guaranteed. (a) Camera response computed using a Macbeth chart. The horizontal axis represents the measured brightness of the Macbeth chart elements and the vertical axis are its tabulated irradiances. Observe that the camera response is highly linear. (b) Projector response (it needs to be linearized). (c) Fall-off of the power light source because of distance. For a uniform light pattern, those lights further from the center of the scene produce a lower radiance. This effect is corrected by reducing the radiance of the brighter sources so as to equalize them to the darker ones.

camera may be considered linear and does not need to be corrected. However, the response of the projector needs to be linearized.

We also need to ensure that there is no angular variation in source brightness with respect to the center of the scene. The main reason for such a variation is that we are generating our sources on a plane rather than a sphere. Since the system has been radiometrically linearized, a single image of the plane taken with a uniform image projected by the projector reveals the fall-off function. Fig. 6.12c depicts this function, which has been discretized according to the resolution that will be used to generate the light patterns. In order to remove this fall-off effect, the radiance of the brighter light sources is reduced so as to equalize them to the darker ones.

With these simple calibrations done, our system satisfies the source and camera linearity assumption we have made.

6.6.3 Relighting static objects

In order to validate our theoretical results and our empirical results with synthetic data, we conducted experiments with static scenes, before moving on to dynamic ones.

Fig. 6.13 shows the objects used for this experiment, a mannequin head and a statue (bust of David). In both cases, a 8×8 grid of patches were used as the individual controllable sources.

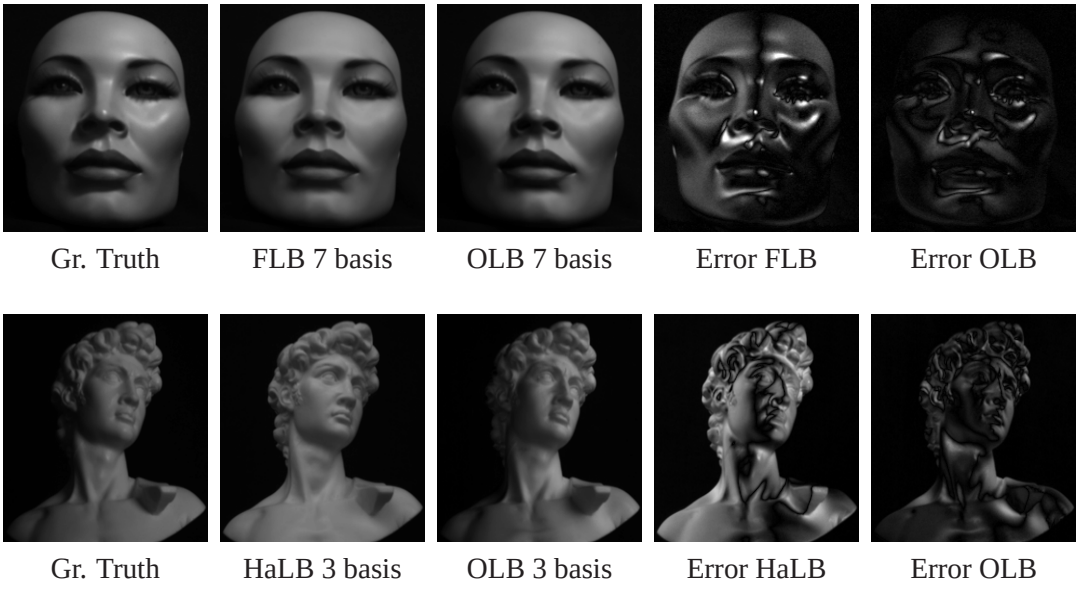


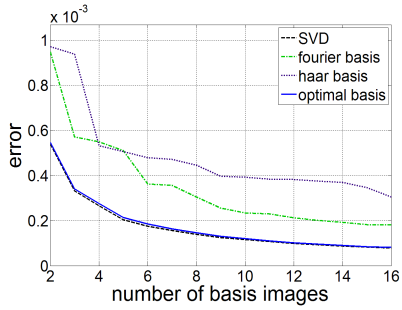
Figure 6.13: **Examples of reconstructed images for the mannequin head and the statue.**

Note that the Lambertian assumption is not valid for both objects as they each have specular components in their reflectance. Even so, due to the additive nature of light (20), the specular reflections can also be reproduced using a linear combination of measurements as long as these reflections do not saturate the camera or produce complex interreflections.

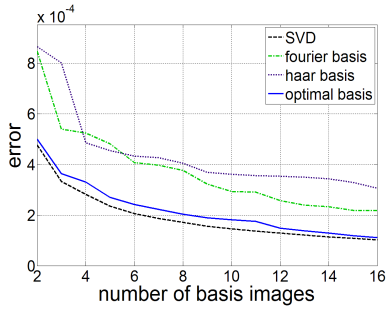
In Fig. 6.13 we show examples of reconstructed images for each of the objects. We see that for the same number of basis, OLB perform better than FLB and HaLB. The differences in the reconstruction quality and error images are clearly visible. In Fig. 6.14, the errors in decoding are plotted as a function of the number of basis images used, for the Fourier, Haar and the optimal lighting bases. As expected, the optimal lighting basis is significantly more efficient than the other bases.

6.6.4 Relighting real moving objects

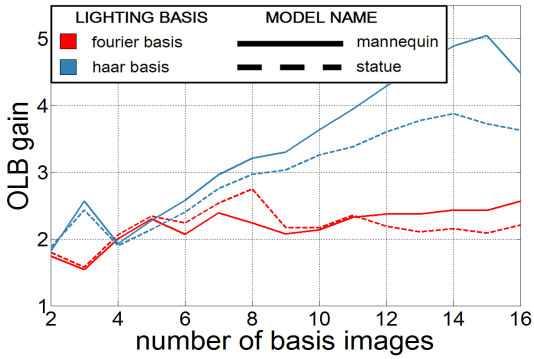
The challenge when relighting moving objects is alignment. Specifically, we need to align points across a window of frames so that when we decode the light patterns we do not blur the information from different points on the object. To do this, we estimate the optical flow over the sequence of images. This problem is made more difficult by the fact that the illumination varies from one frame to the next due to the use of a lighting basis.



(a) Decoding Error for Mannequin Head



(b) Decoding Error for Real Statue

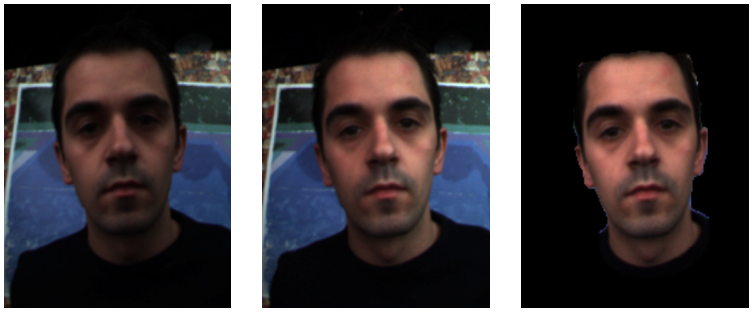


(c) Optimal Lighting Basis Gain

Figure 6.14: **Decoding errors for the synthetic static experiments.** (a) Reconstruction errors for the mannequin in Fig. 6.13 plotted as a function of the number of basis images used, for the Fourier, Haar and the optimal lighting bases. (b) Reconstruction errors for the statue in Fig. 6.13 plotted as a function of the number of basis images used, for the Fourier, Haar and the optimal lighting bases. (c) The gain of the OLB with respect to FLB and HaLB for both experiments, plotted as a function of the number of basis images used.

The alignment may be accomplished by splitting it into two subtasks. Initially a figure background segmentation is performed in order to obtain the correspondence between moving regions in consecutive images. Afterwards, the optical flow of these segmented regions may be computed in order to obtain the correspondence at the subpixel level. Note that if the optical flow were applied to the whole image (and not only to the moving segmented regions), the procedure would be much more complicated and time expensive. By restricting the optical flow algorithm to only small regions of the image highly simplifies the alignment.

Several approaches use alternative techniques apart from 'passive' computer vision algorithms to address each one of these subtasks. For instance, (28) solves the object/background subtraction using infrared light sources and covering the background with special clothes,



(a) (b) (c)

Figure 6.15: **Foreground/Background segmentation.** (a) Image $\mathbf{I}^{t_1}$. (b) Image $\mathbf{I}^{t_2}$. (c) Foreground/background separation in image $\mathbf{I}^{t_2}$. This segmentation is performed adapting the color and contour features from image $\mathbf{I}^{t_1}$. Note that although the two consecutive frames suffer from an abrupt change in appearance (because of the projection of different light patterns), the tracking algorithm proposed in this dissertation allows to obtain a correct foreground/background separation.

which allow an easy and fast figure/background separation. However the use of infrared cameras is constrained to the tracking of heat sources, such as the human body. In order to address the optical flow, some approaches make use of the motion capture technology, where reflective or magnetic markers, which may be easily tracked, are sticked on the surface of the target (47). However, these techniques are invasive and might modify the reflectance properties of the surface. The use of structured light techniques might also simplify the computation of the pixel correpondences throughout time (135; 147). Nevertheless, this is also an invasive technique that might produce some non-desirable artifacts on the image.

In order to obtain a more general approach, not constrained to any type of target, and without requiring from external techniques, we solve both the figure/background segmentation and optical flow by using computer vision algorithms specially tailored to deal with illumination changes in consecutive frames.

The figure/background subtraction is performed by the approach presented in the previous chapter of this dissertation. As we have stated, the use of particle filter formulations to adapt simultaneously the colorspace where the image points are represented, the color distributions of the object and background and the contour of the object, make the method suitable to address the tracking problem in scenarios with abrupt illumination changes, such are the sequences of images illuminated by the light patterns considered in this chapter. Fig. 6.15 shows a result of this segmentation process. Observe that although the abrupt illumination change between two consecutive images (Fig. 6.15a and b), the proposed algorithm is able to address this change

and segment the moving face.

The alignment task is completed by applying an optical flow algorithm to the segmented moving regions, based on a modification of the Lucas-Kanade (73) algorithm. The original version of the algorithm has been adapted in order to increase its robustness to the illumination changes produced by the multiplexed light patterns. Next we give the details of the algorithm:

6.6.4.1 Optical flow using a modification of the Lucas-Kanade algorithm

Our methodology is inspired on the analysis of the algorithm described in (5).

The goal of the Lucas-Kanade algorithm, is to align a template $\mathbf{t}(\mathbf{u})$ to an image $\mathbf{i}(\mathbf{u})$, where $\mathbf{u} = [u, v]^T$ are the pixel coordinates. With the notation of the warping function introduced in Section 3 of the present chapter, the problem may be formulated as the minimization of the sum of the squared error between the template $\mathbf{t}(\mathbf{u})$ and the image $\mathbf{i}(\mathbf{u})$ warped onto the coordinate frame of the template:

$$\Psi_1 = \sum_{\mathbf{u}} [\mathbf{t}(\mathbf{u}) - \mathbf{i}(\Upsilon(\mathbf{u}; \mathbf{q_{it}}))]^2 \tag{6.12}$$

where $\mathbf{q_{it}}$ are the warping parameters mapping from the image to the template.

In this expression, it is assumed that the template $\mathbf{t}(\mathbf{u})$ and the image $\mathbf{i}(\mathbf{u})$ are simply related by a geometric warping. In order to introduce appearance variation in the formulation of the algorithm, Baker et al. (5) consider the minimization of the following function:

$$\Psi_2 = \sum_{\mathbf{u}} [\mathbf{t}(\mathbf{u}) + \sum_{i=1}^{k} \lambda_i \mathbf{a}_i(\mathbf{u}) - \mathbf{i}(\Upsilon(\mathbf{u}; \mathbf{q_{it}}))]^2 \tag{6.13}$$

where $\mathbf{a}_i$ is a set of known appearance variation images, combined with the unknown parameters λ_i, $i = 1, \ldots, k$. This approach has the inconvenience that requires to know a priori the set of images $\mathbf{a}_i$.

Instead, we propose to use a more general minimization function where the changes in the illumination of the template are described by a function Υ_1. The new expression to optimize is:

$$\Psi_3 = \sum_{\mathbf{u}} [\Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1) - \mathbf{i}(\Upsilon_2(\mathbf{u}; \mathbf{q}_2))]^2 \tag{6.14}$$

where the function Υ_1, with the vector of parameters $\mathbf{q}_1$, stands for appearance transformations in the template $\mathbf{t}(\mathbf{u})$, and Υ_2 with the vector of parameters $\mathbf{q}_2$, takes into account the geometric transformations of image $\mathbf{i}(\mathbf{u})$.

To minimize eq. 6.14 we follow a similar optimization procedure than in (5). The minimization is performed simultaneously w.r.t. $\mathbf{q}_1$ and $\mathbf{q}_2$, and the summation is computed over all of the pixels in the template $\mathbf{t}(\mathbf{u})$. Let us assume that the current estimation of $\mathbf{q}_1$ and $\mathbf{q}_2$ is known and then we solve iteratively for increments $\Delta\mathbf{q}_1$ and $\Delta\mathbf{q}_2$. Then, the iterative procedure will include the following two steps:

1. Minimize the function

$$\Psi_3 = \sum_{\mathbf{u}} \left[\Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1 + \Delta\mathbf{q}_1) - \mathbf{i}(\Upsilon_2(\mathbf{u}; \mathbf{q}_2 + \Delta\mathbf{q}_2))\right]^2 \tag{6.15}$$

2. Update parameters

$$\begin{array}{rcl} \mathbf{q}_1 & \leftarrow & \mathbf{q}_1 + \Delta\mathbf{q}_1 \\ \mathbf{q}_2 & \leftarrow & \mathbf{q}_2 + \Delta\mathbf{q}_2 \end{array} \tag{6.16}$$

The two steps are iterated until the convergence of the parameters $\mathbf{q}_1$ and $\mathbf{q}_2$.

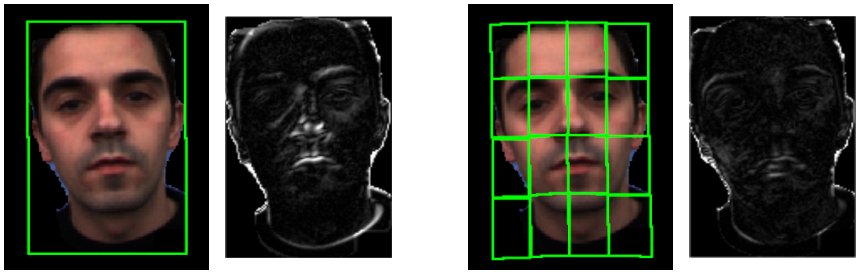
Subsequently, we will derive the optimization of the non-linear Equation 6.15, using a Gauss-Newton gradient descent method. First, the equation is linearized w.r.t. $\mathbf{q}_1$ and $\mathbf{q}_2$ performing a first order Taylor expansion:

$$\begin{aligned} \tilde{\Psi}_3 &= \sum_{\mathbf{u}} \left[\left\{ \Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1) + \frac{\partial \Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1)}{\partial \mathbf{q}_1} \Delta\mathbf{q}_1 \right\} \right. \\ &\quad \left. - \left\{ \mathbf{i}(\Upsilon_2(\mathbf{u}; \mathbf{q}_2)) + \nabla\mathbf{i}(\Upsilon_2(\mathbf{u}; \mathbf{q}_2)) \frac{\partial \Upsilon_2}{\partial \mathbf{q}_2} \Delta\mathbf{q}_2 \right\} \right]^2 \\ &= \sum_{\mathbf{u}} \left[\underbrace{\{\Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1) - \mathbf{i}(\Upsilon_2(\mathbf{u}; \mathbf{q}_2))\}}_{\boldsymbol{\epsilon}(\mathbf{u})} \right. \\ &\quad \left. + \underbrace{\begin{bmatrix} \frac{\partial \Upsilon_1(\mathbf{t}(\mathbf{u}); \mathbf{q}_1)}{\partial \mathbf{q}_1} & \frac{\partial \Upsilon_2}{\partial \mathbf{q}_2} \end{bmatrix}}_{SD(\mathbf{u})} \underbrace{\begin{bmatrix} \Delta\mathbf{q}_1 \\ \Delta\mathbf{q}_2 \end{bmatrix}}_{\Delta\mathbf{q}} \right]^2 \\ &= \sum_{\mathbf{u}} \left[\boldsymbol{\epsilon}(\mathbf{u}) + SD(\mathbf{u})\Delta\mathbf{q}\right]^2 \end{aligned} \tag{6.17}$$

In the previous expressions, $\nabla\mathbf{i} = \left(\frac{\partial \mathbf{i}}{\partial u}, \frac{\partial \mathbf{i}}{\partial v}\right)$ refers to the image gradient, $\boldsymbol{\epsilon}(\mathbf{u})$ is the error image, $SD(\mathbf{u})$ are the *steepest descent* images and $\Delta\mathbf{q}$ is the vector of parameters.

In order to minimize the linearized function $\tilde{\Psi}_3$, we compute its partial derivative w.r.t. $\Delta\mathbf{q}$ and equal to zero:

$$\frac{\partial \tilde{\Psi}_3}{\partial \Delta\mathbf{q}} = 0 \;\Rightarrow\; \sum_{\mathbf{u}} SD(\mathbf{u})^T \left[\boldsymbol{\epsilon}(\mathbf{u}) + SD(\mathbf{u})\Delta\mathbf{q}\right] = 0 \tag{6.18}$$



Pyramid level 1 Pyramid level 3

Figure 6.16: **Pyramidal implementation of the optical flow algorithm.** Left: Pair of aligned template and error image, for the first level of the pyramid. The alignment is performed for the whole template at the same time. Right: Pair of aligned template and error image, for the third level of the pyramid. The alignment is performed independently for each patch of the grid. Observe that different patches have different displacements. This allows to reduce the error, specially on the regions of the nose and the mouth. The error at the contour of the template is produced for slight discrepancies in the figure/ground separation of consecutive frames. This error can be eliminated in a post-processing stage.

Solving eq. 6.18 for $\Delta\mathbf{q}$, we get the closed expression:

$$\Delta\mathbf{q} = -\mathbf{H}^{-1}\sum_{\mathbf{u}} SD(\mathbf{u})^T \boldsymbol{\epsilon}(\mathbf{u}) \tag{6.19}$$

where $\mathbf{H} = \sum_{\mathbf{u}} SD(\mathbf{u})^T SD(\mathbf{u})$ is the *Hessian* matrix.

To summarize, the solution of the algorithm consist of applying iteratively eqs. 6.16 and 6.19 until the convergence of the vector of parameters $\Delta\mathbf{q}$.

For the experimental results that will be shown next, the transformations Υ_1 and Υ_2 have been approximated by affine functions, that is:

$$\begin{aligned}
\Upsilon_1(\mathbf{t}(\mathbf{u});\mathbf{q}_1) &= \begin{bmatrix} 1+q_{11} & q_{12} \end{bmatrix} \begin{bmatrix} \mathbf{t}(\mathbf{u}) \\ 1 \end{bmatrix} \\
\Upsilon_2(\mathbf{u};\mathbf{q}_2) &= \begin{bmatrix} 1+q_{21} & q_{23} & q_{25} \\ q_{22} & 1+q_{24} & q_{26} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}
\end{aligned} \tag{6.20}$$

where $\mathbf{q}_1 = [q_{11}, q_{12}]^T$ and $\mathbf{q}_2 = [q_{21}, q_{22}, \ldots, q_{26}]^T$. Furthermore, in order to handle local deformations of the object being tracked, or to deal with patches of the object that have different change of appearance, we have used a pyramidal implementation. That is, the optical flow is initially applied to the whole area of the object, and subsequently it is applied to small patches, where the initial conditions are given for the results at the previous pyramidal level. Fig. 6.16 shows how with this procedure the alignment error is reduced. Once the alignment is solved, the video sequences are decoded or relighted using the procedure explained in Section 6.3.

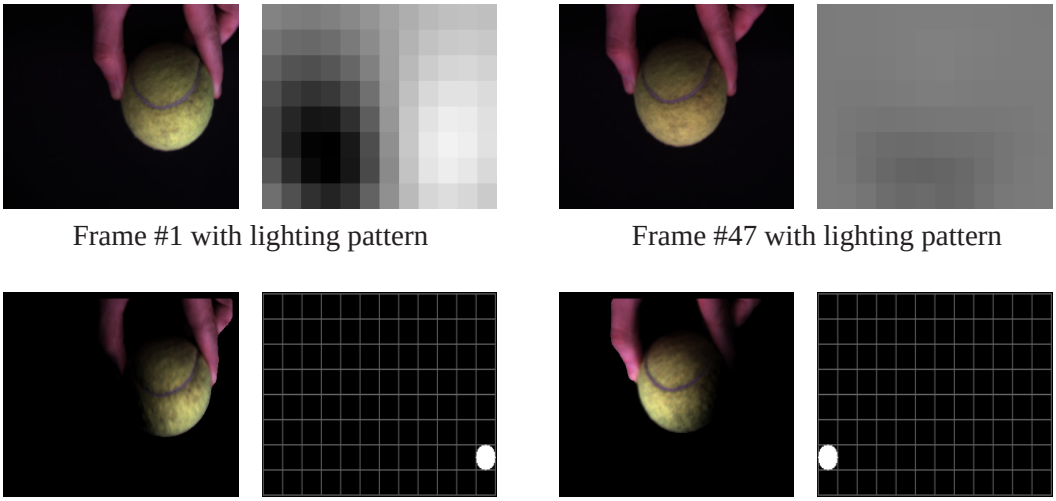


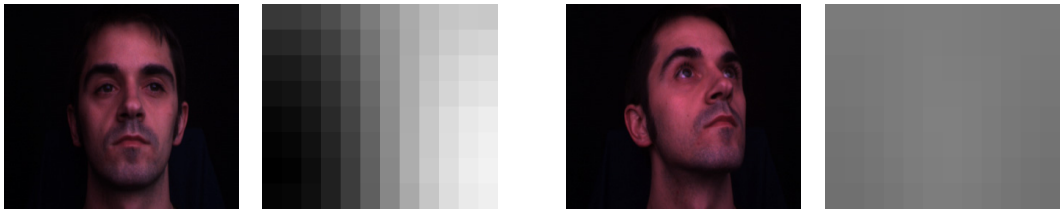
Figure 6.17: **Relighting a tennis ball.** Upper row: two basis image frames of a tennis ball and the lighting pattern used to produce them. Lower row: Relighting results with a single white light source moving in the horizontal axis. The grid in the lighting pattern shows the distribution of the original light sources used to generate the lighting pattern.

6.6.4.2 Relighting results of real video sequences

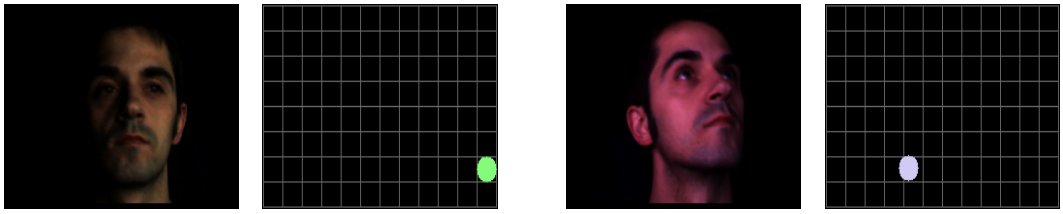
To conclude this section, we will present some relighting results for real video sequences (Figures 6.17, 6.18 and 6.19).

In Fig. 6.17 we show the results of relighting a moving tennis ball. In an off-line procedure, we acquired images of the ball illuminated by single light sources where the sources were arranged in an 8×12 grid on the source plane. Using these images we computed the optimal lighting basis, and the first 3 of these light patterns were used to illuminate the ball while it was moved. The final sequence is acquired at 22fps, and contains 100 frames (the size of the light pattern grid, the frame rate, and the number of optimal lighting basis was the same for all of these experiments). In Fig. 6.17 (bottom row), we show results of relighting the moving ball with a white point light source that moves smoothly across the horizontal axis.

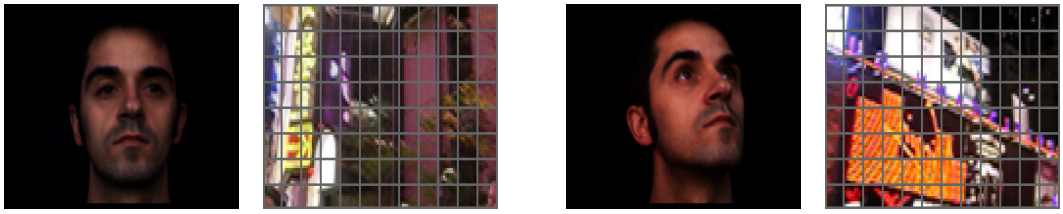
Similar results on the relighting of a human face are shown in Fig. 6.18. In this case, the original sequence has 400 frames. Here, we have included the results of relighting the face with the illumination from New York City's Times Square and the Columbia University campus, which were captured by simply panning a video camera with a wide-angle lens.



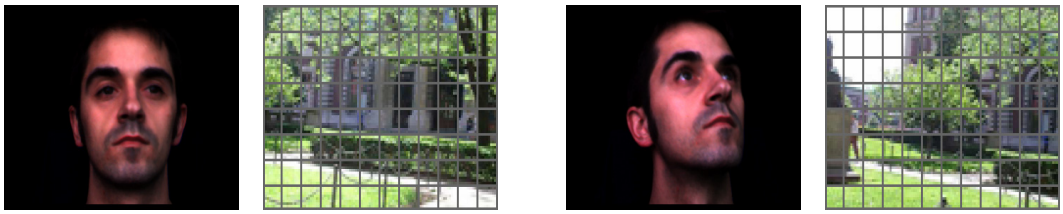
Frame #15 illuminated with reference light | Frame #111 illuminated with reference light



Frame #15 relit with single color light | Frame #111 relit with single color light

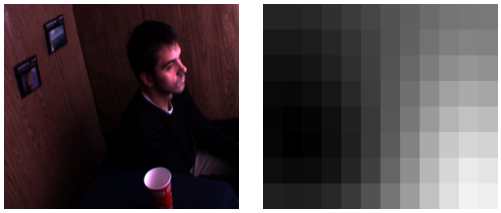


Frame #15 relit with environment light A | Frame #111 relit with environment light A

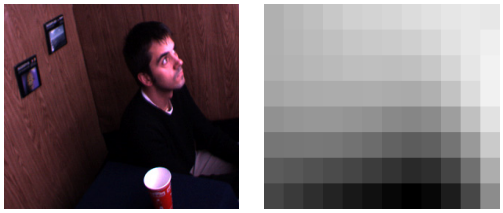


Frame #15 relit with environment light B | Frame #111 relit with environment light B

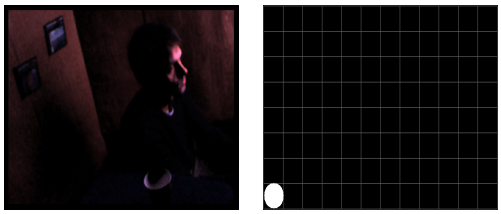
Figure 6.18: **Relighting a moving face with several lighting conditions.** First row: two basis image frames of a human face and lighting patterns used to produce them. The lighting patterns correspond to two of the components of the OLB proposed in this chapter. Second row: Relighting results with a single color light moving in the horizontal axis. Third and fourth rows: Relighting results with lighting from New York City's Times Square and Columbia University Campus, respectively.



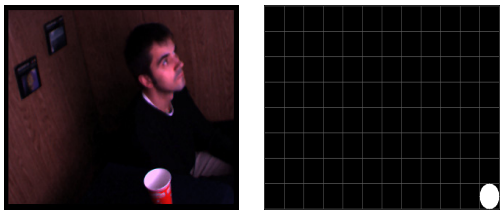
Frame #23 with lighting pattern



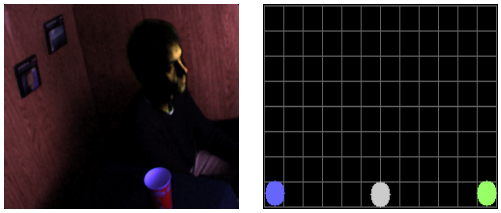
Frame #190 with lighting pattern



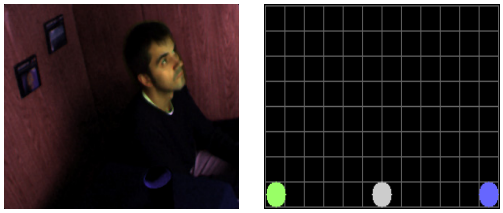
Frame #23 relit with single light source



Frame #190 relit with single light source



Frame #23 locally relit



Frame #190 locally relit

Figure 6.19: **Relighting a corner of a room.** Upper row: two basis image frames of a corner of a room and the lighting patterns used to produce them. Middle row: Relighting results with a single white light source moving in the horizontal axis. Lower row: Local relighting of the scene. The gray light source is a frontal light illuminating the whole scene. The blue light locally relights the cup while the green light focuses on the face.

As we have previously mentioned, one of the advantages of the camera-projector setup that we are using is its scalability. Using this setup we can relight small objects as well as large scenes. In Fig. 6.19, we present results for a room scene with a moving person. In this case, the original sequence has 400 frames. We also use this example to demonstrate that the method may be used for local relighting tasks, where different parts of the scene are lit by different sources. Notice the green light that is focused on the face of the person and the blue light that is shone on the cup. From the cast shadows, one can see that these sources illuminate their respective regions from different directions.

6.7 Summary

Relighting video sequences has recently become an interesting research topic for both computer vision and computer graphics areas. In the same manner than when relighting still images, the reillumination of video sequences may be performed through an image-based approach, where images under new illumination conditions may be generated from linear combinations of a set of basis images, previously acquired under known lighting conditions.

Maybe the most important problem to solve when relighting video sequences, refers to the alignment of the basis images with respect to a common coordinate frame, such that they may be used to generate new relighted frames by just simple linear combinations. This problem is particularly tough because consecutive basis images are illuminated by different light patterns, and thus they suffer from abrupt changes of illumination.

As we have commented in the previous chapter (and proved through a set of experiments), the tracking algorithm proposed in this dissertation is robust to illumination changes. Therefore, we have used this framework to segment and track the moving image regions throughout the video sequence. Subsequently, the result of the figure/background segmentation feeds into an optical flow algorithm which solves the correspondence of these regions at a pixel level.

Apart from the application itself, in this chapter we have also contributed in the study of the lighting basis used to generate the basis images. In particular, we have proposed the use of an *object-dependent* lighting basis which is optimal in the sense that it minimizes the number of reference images that are needed for relighting. The basis is generated off-line, computing SVD over the set of images of the still object illuminated by single light sources. Once the light basis is computed, a subset of the light patterns is used for illuminating the objects in either still scenes or video. Our analysis shows that the lighting basis used here is indeed more effective than other *object independent* lighting bases.

Chapter 7

Conclusions and Future Work

From old movies like “2001: A Space Oddysey” (1968) to most recent films such as “The Matrix” (1999), “Artificial Intelligence” (2001), “Terminator 3” (2003), or “I Robot” (2004), the fantasy of the scriptwriters seems to be endless, and transports us for a pair of hours to futuristic worlds where the mechanical and computational capabilities of robots are astonishing, even outperforming the human being. Unfortunately (or fortunately) this is just science fiction (Fig. 7.1).

The reality is that current research in artificial intelligence, and all its related areas, like robotics or computer vision, are far away from these futuristic worlds, and need to be content by tackling much more humble objectives. A simple task like tracking an object with the eye, which is naturally accomplished for a person, is still an open problem in computer vision. Unpredictable movements of the target, gradual or abrupt changes of illumination, similar objects proximity, cluttered backgrounds, are some of the artifacts that convert the visual tracking in a

Figure 7.1: **This is just science fiction ...** Left and center: Frames from “Terminator 3” (courtesy of Columbia TriStar ©2003 Columbia TriStar). Right: A frame from “I Robot” (courtesy of Fox ©2004 Fox).

really challenging task to be executed by computers.

The work in the present dissertation has addressed this specific problem. We have proposed a tracking framework permitting to deal with disturbances like the just mentioned. The main contributions may be summarized in four major groups. Firstly, we have suggested to use a robust representation of the target color in a colorspace that allows to distinguish the object from the background more clearly than other colorspaces commonly used in literature. Secondly, we have proposed a new probabilistic framework to integrate as many features as necessary, permitting them to mutually interact in order to enhance the estimation of its state. Next, this framework has been utilized to design a real tracking algorithm, validated on several video sequences involving non-linear and non-stationary environments. Finally, a novel video relighting technique has been presented, where the tracking framework designed in this dissertation plays a central role, since the relighting methodology requires to align sequences of images affected by abrupt illumination changes. Furthermore, we have proposed a lighting basis to illuminate the scene, which is optimal for relighting tasks.

In the following sections, the main conclusions and future research for each of the components will be discussed.

7.1 Colorspace representation

An important initial issue that needs to be considered when designing a tracking application refers to the selection of the features to represent the target, allowing to discriminate it from the rest of the scene. Common object cues previously utilized in the literature are shape, motion, geometry or appearance. Within the latter group, color emerges as one of the most utilized features. However, none of the existing approaches paid attention to the selection of the colorspace where image points are represented. Usually, normalized colorspaces have been proposed, based on some criteria of invariance with respect to illumination changes. Nevertheless, this is not the appropriate criterion for a tracking task. In such applications it is more interesting to choose a colorpace maximizing the separability of the target colorpoints with respect to the background colorpoints.

In Chapter 3 we have described the *Fisher colorspace*, which, based on the nonparametric Linear Discriminant Analysis (LDA) (39), computes a plane where the *RGB* colorpoints of the original images are projected, and the distance between the representation of the target and background colorpoints is maximized. The performance of this colorspace was compared

to that of other standard colorspaces, in terms of object/background separability, confirming that the Fisher colorpace is the one that maximizes the distance between both classes. Other interesting properties have been deduced, such as the invariance of the Fisher plane to uniform lighting scalings and translations. However, since we are dealing with moving sequences, the appearance of object and background may dynamically change, requiring the Fisher plane to be updated throughout time. Therefore, it will be estimated like any other object feature.

7.1.1 Future Research

Although the Fisher colorspace has proved to be more effective than other existent colorpaces, its performance still might be improved. Since LDA is a linear technique, when the background and foreground classes are highly overlapped a non-linear technique would be more effective for the separation. In this sense, we leave as part of future work to investigate non-linear separation methods such as the Kernel Principal Component Analysis (KPCA) (111) or the Kernel LDA (KLDA) (69).

7.2 Probabilistic framework for multiple cues integration

Enhance the target representation by using multiple cues has been a common strategy to improve the performance of the tracking techniques. However, most of these algorithms are based on heuristics and ad-hoc rules that only work for specific applications.

In Chapter 5 we have described a new and general probabilistic framework that permits to integrate as many features as necessary, estimated by any algorithm satisfying a hypotheses generation - hypotheses correction scheme and which output is a PDF. This framework allows to integrate both dependent and independent features, and in the special case of dependent features estimated by particle filters, it differs from the partitioned sampling based approaches (76; 77; 142), in the sense that the dependence is considered during the hypotheses correction stage and not during the hypotheses generation phase. This, proves to be much more effective in terms of tracking accuracy and reliability (measured through the survival diagnostic proposed in (77)). Furthermore, since the dynamic model is separately applied to each individual feature and not simultaneously to a global state vector containing all the features, the search area in the configuration space where the hypotheses are formulated is relatively small. As a consequence, in the case that the integrated features are estimated by particle filters, the method proposed here does not suffer from the *curse of dimensionality* problem, which usually affects

particle filter formulations, producing exponential increases in the computation cost when the dimensionality of the state space increases.

7.2.1 Future Research

Note that the proposed framework only considers the integration of multiple cues for a single object tracking. In future research it is planned to extend this formulation to multiple object and multiple cues integration.

7.3 Design of a robust tracking algorithm

In the second part of Chapter 5, the bounding box, colorspace, color distribution and contour features have been integrated in the proposed framework in order to design a robust tracking system that simultaneously accommodates all of the cues. Robustness is provided by the feature integration process as well as by the cue estimation using particle filters. The effectiveness of the method is demonstrated by successfully tracking rigid and non-rigid objects in highly non-stationary environments, which contain cluttered backgrounds and suffer from abrupt illumination changes and non-linear target dynamics. These experiments comprise both synthetic and real video sequences, from indoor and outdoor scenes.

7.3.1 Future Research

More exhaustive analysis might be performed in future work with respect to the following issues:

- **Feature representation models:** Some of the features presented here have been parameterized by simple state vectors that might be improved. For instance, a B-spline model could be used to represent the contour, instead of a discrete set of points distributed along the contour. In the current implementation, the number of Gaussian components approximating the foreground and background color distributions is kept constant, resulting in a color state vector of constant dimensionality. However, new components could be incorporated and some of the components removed from the model if it were necessary.

- **Inclusion of other features:** The representation of the target might be enhanced by including extra features. Texture, depth, motion are some examples of visual cues which could be considered in the future.

- **Dynamic models:** The dynamic models used here when propagating each one of the features, are based on random scalings followed by an addition of Gaussian noise. More accurate dynamic models could be taken into account, for instance second order auto-regressive processes. If a more precise dynamic model is used, the number of samples necessary to approximate the estimation of the features might be reduced.

7.4 Video relighting

A novel and interesting application of the tracking method to video relighting has been described in Chapter 6, which contributes in the following two main topics:

Image alignment under abrupt illumination changes

As we have previously argued, the video relighting is achieved by a linear combination of the pre-acquired sequence frames illuminated under known light patterns, and aligned with respect to a common coordinate frame. The projection of the light patterns on the scene causes abrupt illumination changes in consecutive images of the sequence, and therefore the alignment algorithm needs to be robust to such illumination changes. The first of the contributions of Chapter 6 refers to the use of the proposed tracking algorithm and an adaption of the Lucas-Kanade (73) optical flow algorithm to perform the alignment.

Optimal illumination for video relighting

Although the image alignment is an essential issue to accomphish video relighting, the main contribution of Chapter 6 concerns to the study of the optimal illumination for video relighting.

We have started studying the relighting problem for images of still objects, which is also achieved by linear combination of a set of reference images illuminated under known lighting patterns. In previous approaches, these reference images are most often acquired by either moving a single compact light source over a sphere surrounding the scene, or by sequentially turning on one source at a time among an array of compact sources.

For most relighting-based applications, the goal is to synthesize images of the scene under new illumination conditions such that the synthesized images are as close (in an L^2 sense) to real images as possible. The denser the sampling of the lighting directions for the reference images, the higher the quality of the synthesized images. And one could expect to achieve errorless relighting results (under the assumption of distant light sources), if one were to sample

the space of lighting directions with infinite resolution. Yet, in no case is this practical or even feasible, thus one must settle with the implicit trade-off between quality and the number of reference images.

We have showed in Chapter 6 that it is not simply the number of reference images that determines the quality of relighting, but also the way in which the scene is illuminated. In particular, we showed that the best way (i.e., the way that minimizes the number of reference images) to light the scene is not using a sequence of compact, single point light sources as is most commonly done, but rather to use a sequence chosen from a family, or basis, of lighting patterns each composed of many compact light sources of varying brightness. Furthermore, we showed that the optimal lighting basis can be determined as a simple calibration procedure before acquisition. We demonstrated through experiments on real and synthetic data that the optimal lighting basis significantly reduces the number of reference images that are needed to achieve a desired level of accuracy in the relit images.

This reduction in the number of needed images is particularly critical in the problem of relighting in video as demonstrated by Gardner et al. (41). The reason for this is that the fewer number of reference frames the easiest is the alignment problem commented above. In Chapter 6, we have showed that the optimal lighting basis can reduce the number of light patterns that are needed by a factor of $2-3$ as compared to the spherical harmonic basis used in (41), or other 'object independent' lighting bases such as Fourier or Haar bases.

We have presented several relighting results on real video sequences of moving objects, moving faces, and scenes containing both. In each case although a single video clip was captured, we were able to relight again and again, controlling the lighting direction, extent, and color. In addition, we showed that lighting can be changed over the course of the sequence to produce the effect of a moving source. The lighting could be specified by the user or by some pre-acquired measurement of natural illumination such as an environment map. In the examples presented in this dissertation, we used a video camera with a wide angle lens to acquire a temporally dynamic measurement of the lighting in New York City's Times Square and in the Columbia University campus. These lighting maps were then used to relight one of the video clips of a human face. Finally, we showed that the lighting can even be controlled locally, so that different objects in a scene can be relit in different ways.

7.4.1 Future Research

Yet, challenges remain for improving the performance of the system. In particular, we are investigating ways of using higher frame rate cameras to reduce the relighting errors and allow for faster motion within the video sequences. Furthermore, we are considering the use of adaptive *motion-dependent* relighting bases which adapt based on the motion detected in the scene. For instance, if the target moves fast, a fewer number of basis would be used (in order to simplify the alignment problem), and in case that the target remains static or has low motion, more basis might be used for relighting.

Appendix A

Kalman as a Bayesian Filter

In this Appendix we will derive the Kalman filter equations (Eqs. 2.7 to 2.11) from the Bayesian point of view, instead of the common demonstration based on the minimization of the expected error, as it is done in the original Kalman filter work (58) and in most of the literature of the field (8; 10; 30; 81; 136). Although the Bayesian origin of the filter is also recognized in the majority of the papers, we have not found a clear demonstration, apart from a proof for the one dimensional case in (38). Next we will prove for the general case, that with the assumptions of a linear dynamic model with Gaussian white noise (Eq. 2.5) and Gaussian observation model (Eq. 2.6), the equation of the Bayesian filter (Eq. 2.2) leads to the Kalman filter equations[1].

Initial Assumptions:

$$\begin{aligned} \mathbf{x}^t &= \mathbf{D}^t\mathbf{x}^{t-1} + \mathbf{q}_d^t \qquad \text{with} \quad \mathbf{q}_d^t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_d^t) \\ &\qquad\qquad \text{therefore} \quad p(\mathbf{x}^t|\mathbf{x}^{t-1}) = \mathcal{N}_{\mathbf{x}^t}(\mathbf{D}^t\mathbf{x}^{t-1}, \mathbf{\Sigma}_d^t) \qquad \text{(A.1)} \\ \mathbf{z}^t &= \mathbf{M}^t\mathbf{x}^t + \mathbf{q}_m^t \qquad \text{with} \quad \mathbf{q}_m^t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_m^t) \\ &\qquad\qquad \text{therefore} \quad p(\mathbf{z}^t|\mathbf{x}^t) = \mathcal{N}_{\mathbf{z}^t}(\mathbf{M}^t\mathbf{x}^t, \mathbf{\Sigma}_m^t) \qquad \text{(A.2)} \\ p(\mathbf{x}^t|\mathbf{Z}^t) &= \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_+^t, \mathbf{\Sigma}_+^t) \qquad \text{(A.3)} \end{aligned}$$

Prediction: In the prediction stage, the goal is to compute the likelihood $p(\mathbf{x}^t|\mathbf{Z}^{t-1})$ given the a posteriori PDF in previous iteration $p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})$ and the dynamic model $p(\mathbf{x}^t|\mathbf{x}^{t-1})$, as depicted in Eq. 2.3. Using the relations defined in Eqs. A.1 and A.3, and considering

[1] I would like to thank J.M.Porta-Pleite, for his help in this Section

the linear algebra operations from Appendix B, we have

$$
\begin{aligned}
p(\mathbf{x}^t|\mathbf{Z}^{t-1}) &= \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^t|\mathbf{x}^{t-1})p(\mathbf{x}^{t-1}|\mathbf{Z}^{t-1})d\mathbf{x}^{t-1} \\
\text{(Eqs. A.1, A.3)} &= \int_{\mathbf{x}^{t-1}} \mathcal{N}_{\mathbf{x}^t}(\mathbf{D}^t\mathbf{x}^{t-1}, \boldsymbol{\Sigma}_d^t) \cdot \mathcal{N}_{\mathbf{x}^{t-1}}(\mathbf{x}_+^{t-1}, \boldsymbol{\Sigma}_+^{t-1})d\mathbf{x}^{t-1} \\
\text{(Eq. B.4)} &= \int_{\mathbf{x}^{t-1}} \mathcal{N}_{-\mathbf{D}^t\mathbf{x}^{t-1}}(-\mathbf{x}^t, \boldsymbol{\Sigma}_d^t) \cdot \mathcal{N}_{\mathbf{x}^{t-1}}(\mathbf{x}_+^{t-1}, \boldsymbol{\Sigma}_+^{t-1})d\mathbf{x}^{t-1} \\
\text{(Eq. B.3)} &\propto \int_{\mathbf{x}^{t-1}} \mathcal{N}_{\mathbf{x}^{t-1}}((\mathbf{D}^t)^{-1}\mathbf{x}^t, (\mathbf{D}^t)^{-1}\boldsymbol{\Sigma}_d^t(\mathbf{D}^t)^{-T}) \cdot \mathcal{N}_{\mathbf{x}^{t-1}}(\mathbf{x}_+^{t-1}, \boldsymbol{\Sigma}_+^{t-1})d\mathbf{x}^{t-1} \\
\text{(Eq. B.5)} &\propto \int_{\mathbf{x}^{t-1}} \mathcal{N}_{(\mathbf{D}^t)^{-1}\mathbf{x}^t}(\mathbf{x}_+^{t-1}, (\mathbf{D}^t)^{-1}\boldsymbol{\Sigma}_d^t(\mathbf{D}^t)^{-T} + \boldsymbol{\Sigma}_+^{t-1}) \cdot \mathcal{N}_{\mathbf{x}^{t-1}}(\mathbf{m}_c, \boldsymbol{\Sigma}_c)d\mathbf{x}^{t-1}
\end{aligned}
$$

As the first term of the integral does not depend on $\mathbf{x}^{t-1}$, it can be moved out of the integral:

$$
\begin{aligned}
p(\mathbf{x}^t|\mathbf{Z}^{t-1}) &\propto \mathcal{N}_{(\mathbf{D}^t)^{-1}\mathbf{x}^t}(\mathbf{x}_+^{t-1}, (\mathbf{D}^t)^{-1}\boldsymbol{\Sigma}_d^t(\mathbf{D}^t)^{-T} + \boldsymbol{\Sigma}_+^{t-1}) \cdot \underbrace{\int_{\mathbf{x}^{t-1}} \mathcal{N}_{\mathbf{x}^{t-1}}(\mathbf{m}_c, \boldsymbol{\Sigma}_c)d\mathbf{x}^{t-1}}_{=1} \\
&\propto \mathcal{N}_{(\mathbf{D}^t)^{-1}\mathbf{x}^t}(\mathbf{x}_+^{t-1}, (\mathbf{D}^t)^{-1}\boldsymbol{\Sigma}_d^t(\mathbf{D}^t)^{-T} + \boldsymbol{\Sigma}_+^{t-1}) \\
\text{(Eq. B.3)} &\propto \mathcal{N}_{\mathbf{x}^t}(\mathbf{D}^t\mathbf{x}_+^{t-1}, \boldsymbol{\Sigma}_d^t + \mathbf{D}^t\boldsymbol{\Sigma}_+^{t-1}(\mathbf{D}^t)^T)
\end{aligned}
$$

Therefore, the prediction of the state and covariance, can be expressed as

$$
\begin{aligned}
\mathbf{x}_-^t &= \mathbf{D}^t\mathbf{x}_+^{t-1} && \text{(A.4)} \\
\boldsymbol{\Sigma}_-^t &= \boldsymbol{\Sigma}_d^t + \mathbf{D}^t\boldsymbol{\Sigma}_+^{t-1}(\mathbf{D}^t)^T && \text{(A.5)}
\end{aligned}
$$

which correspond to the predicted state and predicted covariance equations of the Kalman filter (Eqs. 2.7 and 2.8).

Correction: In the correction stage, the predicted distribution $p(\mathbf{x}^t|\mathbf{Z}^{t-1}) = \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_-^t, \boldsymbol{\Sigma}_-^t)$ and the observation density $p(\mathbf{z}^t|\mathbf{x}^t) = \mathcal{N}_{\mathbf{z}^t}(\mathbf{M}^t\mathbf{x}^t, \boldsymbol{\Sigma}_m^t)$ are combined according to Eq. 2.4 in order to obtain the a posteriori estimates of the state and covariance:

$$
\begin{aligned}
p(\mathbf{x}^t|\mathbf{Z}^t) &\propto p(\mathbf{z}^t|\mathbf{x}^t)p(\mathbf{x}^t|\mathbf{Z}^{t-1}) \\
&\propto \mathcal{N}_{\mathbf{z}^t}(\mathbf{M}^t\mathbf{x}^t, \boldsymbol{\Sigma}_m^t) \cdot \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_-^t, \boldsymbol{\Sigma}_-^t) \\
\text{(Eq. B.4)} &\propto \mathcal{N}_{-\mathbf{M}^t\mathbf{x}^t}(-\mathbf{z}^t, \boldsymbol{\Sigma}_m^t) \cdot \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_-^t, \boldsymbol{\Sigma}_-^t) \\
\text{(Eq. B.3)} &\propto \mathcal{N}_{\mathbf{x}^t}((\mathbf{M}^t)^{-1}\mathbf{z}^t, (\mathbf{M}^t)^{-1}\boldsymbol{\Sigma}_m^t(\mathbf{M}^t)^{-T}) \cdot \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_-^t, \boldsymbol{\Sigma}_-^t) \\
\text{(Eq. B.5)} &\propto \mathcal{N}_{(\mathbf{M}^t)^{-1}\mathbf{z}^t}(\mathbf{x}_-^t, (\mathbf{M}^t)^{-1}\boldsymbol{\Sigma}_m^t(\mathbf{M}^t)^{-T} + \boldsymbol{\Sigma}_-^t) \cdot \mathcal{N}_{\mathbf{x}^t}(\mathbf{m}_c, \boldsymbol{\Sigma}_c)
\end{aligned}
$$

Since the first term is not a function of $\mathbf{x}_t$ we can consider it as a proportional factor. As a consequence, the a posteriori distribution is reduced to:

$$p(\mathbf{x}^t|\mathbf{Z}^t) \propto \mathcal{N}_{\mathbf{x}^t}(\mathbf{m}_c, \mathbf{\Sigma}_c) = \mathcal{N}_{\mathbf{x}^t}(\mathbf{x}_+^t, \mathbf{\Sigma}_+^t)$$

where the terms $\mathbf{x}_+^t$ and $\mathbf{\Sigma}_+^t$ are expressed according to Eqs. B.6 and B.7 respectively. Rearranging the terms, it can be showed that these expressions correspond precisely to the equations 2.9, 2.10, and 2.11 of the Kalman filter in the correction stage. Let us start rearranging the components of the covariance term:

$$\begin{aligned}
\mathbf{\Sigma}_+^t &= ((\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}\mathbf{M}^t + (\mathbf{\Sigma}_-^t)^{-1})^{-1} \\
\text{(Eq. B.1)} &= \mathbf{\Sigma}_-^t - \underbrace{\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t + \mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T)^{-1}}_{\mathbf{K}^t}\mathbf{M}^t\mathbf{\Sigma}_-^t) \\
&= \mathbf{\Sigma}_-^t - \mathbf{K}^t\mathbf{M}^t\mathbf{\Sigma}_-^t \qquad \text{(A.6)}
\end{aligned}$$

where

$$\mathbf{K}^t = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t + \mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T)^{-1} \qquad \text{(A.7)}$$

is the Kalman gain. Note that Eqs. A.6 and A.7 correspond to the Kalman covariance update rule (Eq. 2.10) and Kalman gain (Eq. 2.11), respectively.

With respect to the state vector term, we have:

$$\begin{aligned}
\mathbf{x}_+^t &= \mathbf{\Sigma}_+^t((\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}\mathbf{M}^t(\mathbf{M}^t)^{-1}\mathbf{z}^t + (\mathbf{\Sigma}_-^t)^{-1}\mathbf{x}_-^t) \\
&= \mathbf{\Sigma}_+^t((\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}\mathbf{z}^t + (\mathbf{\Sigma}_-^t)^{-1}\mathbf{x}_-^t) \\
&= (\mathbf{\Sigma}_-^t - \mathbf{K}^t\mathbf{M}^t\mathbf{\Sigma}_-^t)((\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}\mathbf{z}^t + (\mathbf{\Sigma}_-^t)^{-1}\mathbf{x}_-^t) \\
&= \mathbf{x}_-^t - \mathbf{K}^t\mathbf{M}^t\mathbf{x}_-^t + \left(\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} - \mathbf{K}^t\mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}\right)\mathbf{z}^t
\end{aligned} \qquad \text{(A.8)}$$

We can show that:

$$\mathbf{K}^t = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} - \mathbf{K}^t\mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} \qquad \text{(A.9)}$$

Proof:

$$\begin{aligned}
&\mathbf{K}^t = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} - \mathbf{K}^t\mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} \Leftrightarrow \\
&\mathbf{K}^t(\mathbb{I} + \mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}) = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1} \Leftrightarrow \\
&\mathbf{K}^t = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1}(\mathbb{I} + \mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t)^{-1})^{-1} \Leftrightarrow \\
&\mathbf{K}^t = \mathbf{\Sigma}_-^t(\mathbf{M}^t)^T(\mathbf{\Sigma}_m^t + \mathbf{M}^t\mathbf{\Sigma}_-^t(\mathbf{M}^t)^T)^{-1}
\end{aligned}$$

which is true according to the previous definition of the Kalman gain (Eqs. 2.11 and A.7). Therefore, equation A.8 can be shortened to

$$\begin{aligned} \mathbf{x}_+^t &= \mathbf{x}_-^t - \mathbf{K}^t\mathbf{M}^t\mathbf{x}_-^t + \mathbf{K}^t\mathbf{z}^t \\ &= \mathbf{x}_-^t - \mathbf{K}^t(\mathbf{M}^t\mathbf{x}_-^t + \mathbf{z}^t) \end{aligned} \tag{A.10}$$

that is the same expression as the state update rule in the Kalman filter formulation (Eq. 2.9).

Appendix B

Mathematical Formulas

B.1 Matrix Relations

Formulae obtained from (18).

The Woodbury indentity Inverse :

$$(\mathbf{A}+\mathbf{CBC}^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{C}(\mathbf{B}^{-1}+\mathbf{C}^T\mathbf{A}^{-1}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{A}^{-1} \tag{B.1}$$

B.2 Normal Densities

Formulae obtained from (18).

Normal density PDF: The density of $\mathbf{x} \sim \mathcal{N}(\mathbf{m}, \mathbf{\Sigma})$ is

$$p(\mathbf{x}) = \frac{1}{\sqrt{\det(2\pi\mathbf{\Sigma})}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{m})\right] \tag{B.2}$$

Rearranging Means:

$$\mathcal{N}_{\mathbf{Ax}}(\mathbf{m}, \mathbf{\Sigma}) \propto \mathcal{N}_{\mathbf{x}}(\mathbf{A}^{-1}\mathbf{m}, \mathbf{A}^{-1}\mathbf{\Sigma}\mathbf{A}^{-T}) \tag{B.3}$$

$$\mathcal{N}_{\mathbf{x}}(\mathbf{Am}, \mathbf{\Sigma}) = \mathcal{N}_{-\mathbf{Am}}(-\mathbf{x}, \mathbf{\Sigma}) \tag{B.4}$$

Product of Gaussian densities:

$$\mathcal{N}_{\mathbf{x}}(\mathbf{m_1}, \mathbf{\Sigma_1}) \cdot \mathcal{N}_{\mathbf{x}}(\mathbf{m_2}, \mathbf{\Sigma_2}) = \mathcal{N}_{\mathbf{m}_1}(\mathbf{m}_2, (\mathbf{\Sigma}_1+\mathbf{\Sigma}_2)) \cdot \mathcal{N}_{\mathbf{x}}(\mathbf{m_c}, \mathbf{\Sigma_c}) \tag{B.5}$$

where:

$$\mathbf{m}_c = (\mathbf{\Sigma}_1^{-1}+\mathbf{\Sigma}_2^{-1})^{-1}(\mathbf{\Sigma}_1^{-1}\mathbf{m}_1+\mathbf{\Sigma}_2^{-1}\mathbf{m}_2) \tag{B.6}$$

$$\mathbf{\Sigma_c} = (\mathbf{\Sigma}_1^{-1}+\mathbf{\Sigma}_2^{-1})^{-1} \tag{B.7}$$

Bibliography

[1] M.A.Abidi, "Fusion of multi-dimensional data using regularization", *Data Fusion in Robotics and Machine Intelligence, M.A.Abidi and R.C.Gonzalez (Eds.), Academic Press*, pp.415-455, 1992. 4.2

[2] A.Arthur, "Object tracking through adaptive cue integration", *In MSc thesis, Artificial Intelligence, School of Informatics, University of Edinburgh*, 2004. 4.4.2

[3] S.Arulampalam, S.Maskell, N.J.Gordon, T.Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking", *IEEE Transactions of Signal Processing*, Vol.50(2), pp.174-188, 2002. 2.2, 2.3, 2.3

[4] Y.Azoz, L.Devi, R.Sharma, "Reliable tracking of human arm dynamics by multiple cue integration and constraint fusion", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.905-910, 1998. 4.3, 4.3.2.1, 4.4.3

[5] S.Baker, I.Matthews, "Lucas-kanade 20 years on: A unifying framework", *International Journal of Computer Vision*, Vol.56(3), pp.221-255, 2004 6.6.4.1, 6.6.4.1, 6.6.4.1

[6] K.Barnard, B.Funt, V.Cardei, "A comparison of computational color constancy algorithms, Part one; Theory and experiments with synthetic data", *IEEE Transactions on Image Processing*, Vol.11(9), pp.972-984, 2002. 3.2.4

[7] K.Barnard, B.Funt, V.Cardei, "A comparison of computational color constancy algorithms, Part 2; Experiments with images", *IEEE Transactions on Image Processing*, Vol.11(9), pp.985-996, 2002. 3.2.4

[8] Y.Bar-Shalom, T.Fortmann, "Tracking and data association", *Academic Press*, 1988. 2.2, 2.2, 4.3.2.3, A

[9] Y.Bar-Shalom, X.R.Li, "Multitarget-multisensor tracking: Principles and techniques", *Storrs, CT: YBS*, 1995. 4.1

[10] Y.Bar-Shalom, X.R.Li, T.Kirubarajan, "Estimation with applications to tracking and navigation", *John Wiley & Sons, New York*, 2001. 2.2, A

[11] T.Bayes, "An essay towards solving a problem inthe doctrine of chances", *Philosophical Transactions of the Royal Society*, Vol.53, pp.370-418, 1783. 2.1, 4.2

[12] D.Beymer, K.Konolige, "Real-time tracking of multiple people using continuous detection", *Framerate Workshop (in conjunction with ICCV)*, 1999. 4.3, 4.3.2.1, 4.4.3

[13] S.Birchfield, "An elliptical head tracker", *Proc. Asilomar Conference on Signals, Systems and Computers*, 1997. 3.1, 3.3

[14] S.Birchfield, "Elliptical head tracking using intensity gradients and color histograms", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.232-237, 1998. 4.3, 4.3.1.1, 4.4.3

[15] C.Bishop, "Neural networks for pattern recognition", *Oxford University Press*, 1995. 3.3.2

[16] M.Black, P.Anandan, "The robust estimation of multiple motions: parametric and piecewise-smooth flow fields", *Computer Vision and Image Understanding*, Vol.63(1), pp.75-104, 1996. 3.1

[17] I.Block, "Information combination operators for data fusion: A comparative review with classification", *IEEE Transactions on Systems, Man, And Cybernetics, Part A: systems ans humans*, Vol.26(1), pp.42-52, 1996. 4.2

[18] K.Brandt, M.Syskind, "The matrix cookbook", *http://2302.dk/uni/matrixcookbook.html*, 2005. B.1, B.2

[19] C.Bräutigam, J.O.Eklundh, H.I.Christensen, "A model-free voting approach for integrating multiple cues", *Proc. European Conference on Computer Vision (ECCV)*, 1998. 4.3, 4.3.1.2

[20] I.Busbridge, "The Mathematics of Radiative Transfer", *Cambridge University Press*, 1960. 6.1, 6.3.1, 6.6.3

[21] J.Canny, “A computational approach to edge detection”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.8(6), pp.679-698, 1986. 3.4.1

[22] J.Carpenter, P.Clifford, P.Fearnhead, “Improved particle filter for nonlinear problems”, *IEEE Proceedings on Radar and Sonar Navigation*, Vol.146, 1999. 2.2, 2.3

[23] J.J.Clark, A.L.Yuille, “Data fusion for sensory information Processing Systems”, *Kluwer Academic Publishers*, 1990. 1.2, 4.2, 4.5, 5.1

[24] T.F.Cootes, G.J.Edwards, C.J.Taylor, “Active Appearance Models”, *Proc. European Conference on Computer Vision (ECCV)*, Vol.2, pp.484-498, 1998. 4.3, 4.3.2.2, 4.4.1, 4.4.3

[25] J.Crowley, F.Berard, “Multi-modal tracking of faces for video communications”, *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.640-645, 1997. 3.2.2, 4.3, 4.3.1.1, 4.3.2.1, 4.4.2, 4.4.3

[26] T.Darrel, G.Gordon, J.Woodfill, M.Harville, “A virtual mirror interface using real-time robust face tracking”, *Proc. International Conference on Automatic Face and Gesture Recognition*, pp.616-621, 1998. 4.3, 4.3.2.1

[27] P.Debevec, T.Hawkins, C.Tchou, H.Duiker, W.Sarokin, “Acquiring the reflectance field of a human face”, *Proc. of SIGGRAPH, ACM*, pp.145-156, 2000. 6.1, 6.2, 6.3.1

[28] P.Debevec, A.Wenger, C.Tchou, A.Gardner, J.Waese, T.Hawkins, “A lighting reproduction approach to live-action composing”, *Proc. of SIGGRAPH, ACM*, vol.21(3), pp.547-556, 2002. 6.1, 6.2, 6.6.4

[29] A.Dempster, N.Laird, D.Rubin, “Maximum likelihood estimation from incomplete data via the EM algorithm”, *Journal of the Royal Statistical Society B*, Vol.39, pp.1-38, 1977. 3.3.2

[30] E.D.Dickmanns, “Recursive state estimation”, *Lecture Notes, California Institute of Technology*, 1996. 2.2, A

[31] J.Dorsey, F.Sillion, D.Greenberg, “ Design and simulation of opera lighting and projection effects”, *Proc. of SIGGRAPH, ACM*, pp.41-50, 1991. 6.2

[32] A.Doucet, N.de Freitas, N.Gordon, editors, “Sequential Monte Carlo in practice”, *Springer-Verlag, New York*, 2001. 2.2, 2.3

[33] R.O.Duda, P.E.Hart, D.G.Stork, "Pattern classification", *John Wiley & Sons, 2nd Edition, New York*, 2001. 3.2.3, 3.2.3

[34] G.D.Finlayson, B.V.Funt, K.Barnard, "Color constancy under varying illumination", *Proc. International Conference on Computer Vision (ICCV)*, pp.720-725, 1995. 3.2.4

[35] M.A.T.Figueiredo, A.K.Jain, "Unsupervised learning of finite mixture models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.24(3), 2002. 3.3.2, 3.11, 5.4.3

[36] G.D.Finlayson, B.Schiele, J.L.Crowley, "Comprehensive colour image normalization", *Proc. European Conference on Computer Vision (ECCV)*, Vol.1, pp.475-490, 1998. 3.2.5

[37] J.Foley, A.van Dam, S.Feiner, J.Hughes, "Computer graphics: principles and practice", *Addison-Wesley*, 1990. 3.2.1

[38] D.A.Forsyth, J.Ponce, "Computer vision, A modern approach", *Prentice Hall*, 2003. 3.2, A

[39] K.Fukunaga, "Introduction to statistical pattern recognition", *Academic Press*, 1990. 3.2.3, 3.2.3, 3.2.3, 2, 5.2.1, 7.1

[40] A.Gelb (Editor), "Applied optimal estimation", *MIT Press, Cambridge*, 1974. 2.2

[41] A.Gardner, C.Tchou, A.Wenger, T.Hawkins, P.Debevec, "Postproduction re-illumination of live action using time-multiplexed lighting", *SIGGRAPH (Poster), ACM*, 2004. 6.2, 7.4

[42] A.Georghiades, P.Belhumeur, D.Kriegman, " From few to many: Illumination cone models for face recognition under variable lighting and pose", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.23(6), pp.643-660, 2001. 6.1

[43] A.Georghiades, D.Kriegman, P.Belhumeur, "Illumination cones for recognition under variable lighting: Faces", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.52-59, 1998. 6.2

[44] W.E.L.Grimson, T.Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data", *International Journal of Robotics Research*, Vol.3(3), pp.3-35, 1984. 4.1

[45] G.D.Hager, P.N.Belhumeur, "Efficient region tracking with parametric models of geometry and illumination", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.20(10), pp.1025-1039, 1998. 4.3, 4.3.2.2, 4.4.1, 4.4.2, 4.4.3

[46] P.Hallinan, "A low-dimensional representation of human faces for arbitrary lighting conditions", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.995-999, 1994. 6.2

[47] T.Hawkins, A.Gardner, C.Tchou, F.Goransson, P.Debevec, "Animatable facial reflectance fields", *Proc. Eurographics Symposium on Rendering*, 2004. 6.1.2, 6.2, 6.6.4

[48] E.Hayman, J.O.Eklundh, "Probabilistic and Voting Approaches to Cue Integration for Figure-Ground Segmentation", *Proc. European Conference on Computer Vision (ECCV)*, pp.469-486, 2002. 3.2.2, 4.2, 4.2, 4.3, 4.3.1.1, 4.3.1.3, 4.4.1, 4.4.2

[49] M.Irani, P.Anandan, "All about direct methods", *Proc. International Workshop on Vision Algorithms, Springer Verlag*, LNCS 1883, pp.267-277, 2000. 3.1

[50] M.Isard, "Visual motion analysis by probabilistic propagation of conditional density", *PhD Thesis, University of Oxford*, 1998. 2.2, 2.3, 2.3

[51] M.Isard, A.Blake, "CONDENSATION-Conditional Density Propagation for Visual Tracking Michael", *International Journal of Computer Vision*, Vol.29(1), pp.5-28, 1998. 2.2, 2.3, 3.1, 4.3, 4.3.2.1, 4.3.2.3, 4.4.3, 5.3, 5.3.1, 5.6.1

[52] M.Isard, A.Blake, "ICondensation: Unifying low level and high-level tracking in a stochastic framework", *Proc. European Conference on Computer Vision (ECCV)*, pp.893-908, 1998. 4.3, 4.3.2.1, 5.6.1

[53] R.S.Jadon, S.Chaudhury, K.K.Biswas, "A fuzzy theoretic approach for video segmentation using synthactic features", *Pattern Recognition Letters*, Vol.22, pp.1359-1369, 2001. 4.3, 4.3.1.4

[54] T.S.Jebara, A.Pentland, "Parameterized structure from motion for 3D adaptive feedback tracking of faces", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.144-150, 1997. 3.3, 3.3.2

[55] F.V.Jensen, "An introduction to Bayesian Networks", *Springer-Verlag*, 1996. 4.3.1.3

[56] M.J.Jones, J.M.Rehg, "Statistical color models with application to skin detection", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.274-280, 1999. 3.3

[57] S.Julier, A.Skewed, "A skewed approach to filtering", *Proc. Society of Photo-Optical Instrumentation Engineers (SPIE)*, Vol.3373, pp.271-282, 1998. 2.2

[58] R.E.Kalman, "A new approach to linear filtering and prediction Problems", *Transactions of the ASME-Journal of Basic Engineering*, pp.35-45, 1960. 2.2, A

[59] M.Kass, A.Witkin, D.Terzopoulos,"Snakes: Active contour models", *International Journal of Computer Vision*, Vol.1, pp.321-331, 1987. 3.4.1, 5.5.6

[60] Z.Khan, T.Balch, F.Dellaert, "A rao-blackwellized particle filter for eigentracking", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.980-986, 2004. 4.3, 4.3.2.3, 4.4.1, 4.4.2, 4.4.3, 4.5, 5.3.1

[61] S.Khan, M.Shah, "Object based segmentation of video using color, motion and spatial infomation", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol.2, pp.746-751, 2001. 3.2.2, 4.3, 4.3.1.1

[62] L.Klein, "Sensor and data fusion concepts and applications", *Society of Photo-Optical Instrumentation Engineers (SPIE)*, 1993. 4.1

[63] M.Koudelka, P.Belhumeur, S.Magda, D.Kriegman, "Image-based modeling and rendering of surfaces with arbitrary brdfs", *Proceedings of International Conference on Computer Vision (ICCV)*, pp.568-575, 2001. 6.2

[64] D.Kragić, H.Christensen, "Cue Integration for visual servoing", *IEEE Transactions on Robotics and Automation*, Vol.17(1), pp.18-27, 2001. 3.2.2, 4.3, 4.3.1.2, 4.3.1.4, 4.4.1, 4.4.3

[65] H.Kruppa, B.Schiele, "Hierarchical combination of object models using mutual information", *Proc. British Machine Vision Conference (BMCV)*, 2001. 4.3, 4.3.2.1

[66] J.J.Kuch, T.S.Huang, "Vision-based hand modeling and tracking for virtual teleconferencing and telecollaboration", *Proc. International Conference on Computer Vision (ICCV)*, pp.666-671, 1995. 3.1

[67] I.Leichter, M.Lindenbaum, E.Rivlin, "A probabilistic framework for combining tracking algorithms", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol.2, pp.445-451, 2004. 4.3, 4.3.1.3, 4.3.2.3, 4.4.1, 5.1

[68] H.Lensch, J.Kautz, M.Goesele, W.Heidrich, H.Seidel, "Image-based reconstruction of spatial appearance and geometric detail", *ACM Transactions on Graphics*, Vol.22, pp.234-257, 2003. 6.1

[69] Y.Li, S.Gong, H.Liddell, "Recognising trajectories of facial identities using Kernel Discriminant Analysis", *Proc. British Machine Vision Conference (BMVC)*, pp.613-622, 2001. 7.1.1

[70] Z.Lin, T.Wong, H.Shum, "Relighting with the reflected irradiance field: Representation,sampling and reconstruction", *International Journal of Computer Vision*, Vol.49, pp.229-246, 2002. 6.1

[71] J.S.Liu, "Monte Carlo strategies in scientific computing", *Springer-Verlag: New York*, 2003. 2.2, 2.3

[72] P.Lombardi, "A study on data fusion techniques for vision modules", *Technical Report, Università di Pavia*, 2002. 4.2

[73] B.Lucas, T.Kanade, "An iterative image registration technique with an application to stereo vision", *Proceedings of the International Joint Conference on Artificial Intelligence*, pp.674-679, 1981. 6.1.2, 6.6.4, 7.4

[74] R.C.Luo, M.G.Kay, "Multisensor integration and fusion in intelligent systems", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.19(5), pp.901-931, 1989. 4.1, 4.2

[75] J.MacCormick, "Probabilistic modeling and stochastic algorithms for visual localisation and tracking", *PhD Thesis, University of Oxford*, 2000. 2.2, 2.3, 2.3, 3.1, 4.3.2.3, 5.3.1, 5.6

[76] J.MacCormick, A.Blake,"Probabilistic exclusion and partitioned sampling for multiple object tracking", *International Journal of Computer Vision*, Vol.39(1), pp.57-71, 2000. 4.3, 4.3.2.3, 4.4.3, 5.1, 5.3.1, 5.3.1, 5.3.1, 7.2

[77] J.MacCormick, M.Isard,"Partitioned sampling, articulated objects, and interface-quality hand tracking", *Proc. European Conference on Computer Vision (ECCV)*, Vol.2, pp.3-19, 2000. 4.3, 4.3.2.3, 4.4.3, 5.1, 5.3.1, 7.2

[78] T.Malzbender, D.Gelb, H.Wolters, "Polynomial texture maps", *Proc. of SIGGRAPH, ACM*, pp.519-528, 2001. 6.2

[79] W.Matusik, H.Pfister, A.Ngan, P.Beardsley, R.Ziegler, L.McMillan, "Image-based 3d photography using opacity hulls", *Proc. of SIGGRAPH, ACM*, pp. 427-437, 2002. 6.2

[80] I.Matthews, S.Baker, "Active appearance models revisited", *International Journal of Computer Vision*, Vol.60(2), pp.135-164, 2004. 4.3, 4.3.2.2, 4.4.1, 4.4.3

[81] P.S.Maybeck, "Stochastic models, estimation and control", *Academic Press, New York*, Vol.1, 1979. 2.2, A

[82] T.B.Moeslund, E.Granum, "A survey of computer vision-based human motion capture", *Computer Vision and Image Understanding*, Vol.81, pp.231-268, 2001. 4.5

[83] P.Del Moral, "Non-linear filtering: Interacting particle solution", *Markov processes and related fields*, Vol.2(4), pp.555-581, 1996. 2.2, 2.3

[84] F.Moreno-Noguer, S.K.Nayar, P.N.Belhumeur, "Optimal Illumination for Image and Video Relighting (full paper)", *Proc. IEE European Conference on Visual Media Production (CVMP)*, 2005.

[85] F.Moreno-Noguer, A.Sanfeliu, D.Samaras, "Integration of Conditionally Dependent Object Features for Robust Figure/Background Segmentation", *International Conference on Computer Vision (ICCV)* , 2005. 4.3, 4.3.2.3, 4.4.2, 4.4.3

[86] F.Moreno-Noguer, S.K.Nayar, P.N.Belhumeur, "Optimal Illumination for Image and Video Relighting (short sketch)", *SIGGRAPH (Sketch) ,ACM*, 2005.

[87] F.Moreno-Noguer, A.Sanfeliu, "A Framework to Integrate Particle Filters for Robust Tracking in Non-stationary Environments", *Proc. Iberian Conference on Pattern Recognition and Image Analysis, (IBPRIA), LNCS 3522*, 2005. 4.3.2.3

[88] F.Moreno-Noguer, A.Sanfeliu, "Integration of Shape and a Multihypothesis Fisher Color Model for Figure-Ground Segmentation in Non-Stationary Environments", *Proc. International Conference on Pattern Recognition (ICPR)*, Vol.4, pp.771-774, 2004. 4.3, 4.3.2.1

[89] F.Moreno-Noguer, A.Sanfeliu, D.Samaras, "Fusion of a Multiple Hypotheses Color Model and Deformable Contours for Figure Ground Segmentation in Dynamic Enviroments", *Proc. Workshop on Articulated and Non-Rigid Motion (in conjunction with CVPR)*, 2004.

[90] F.Moreno-Noguer, A.Sanfeliu, "Adaptative Color Model for Figure Ground Segmentation in Dynamic Environments", *Proc. Iberoamerican Congress on Pattern Recognition (CIARP)*, pp.37-44, 2004.

[91] F.Moreno-Noguer, J.Andrade-Cetto, A.Sanfeliu, "Fusion of Color and Shape for Object Tracking under Varying Illumination", *Proc. Iberian Conference on Pattern Recognition and Image Analysis (IBPRIA), LNCS 2652*, pp.580-588, 2003. 3.3

[92] F.Moreno-Noguer, A.Tarrida, J.Andrade-Cetto, A.Sanfeliu, "3D Real Time Head Tracking Fusing Color Histograms and Stereovision", *Proc. International Conference on Pattern Recognition (ICPR)*, Vol.1, pp.368-371, 2002. 4.3, 4.3.2.1

[93] F.Moreno-Noguer, "Pattern recognition systems", *Final Year Project for the Electrical Engineering Degree, University of Barcelona*, 2002.

[94] F.Moreno-Noguer, J.Andrade-Cetto, A.Sanfeliu, "Localization of Human Faces Fusing Color Segmentation and Depth from Stereo", *Proc. IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp.527-536, 2001.

[95] F.Moreno-Noguer, "Development of a stereo vision system for a mobile robot", *Final Year Project for the Industrial Engineering Degree, Technical University of Barcelona*, 2001.

[96] S.Nayar, P.Belhumeur, T.Boult, "Lighting sensitive display", *ACM Transactions on Graphics*, Vol.23(4), pp.963-979, 2004. 6.2

[97] J.Nimeroff, E.Simoncelli, J.Dorsey, "Efficient re-rendering of naturally illuminated environments", *Proc. Eurographics Workshop on Rendering*, pp.359-373, 1994. 6.1, 6.2, 6.3.1

[98] K.Nishino, S.Nayar, "Eyes for relighting", *ACM Transactions on Graphics*, Vol.23(3), pp.704-711, 2004. 6.1, 6.2

[99] P.Nordlund, J.O.Eklundh, "Towards a seeing agent", *Proc. International Workshop on Cooperative distributed Vision*, pp.93-123, 1997. 4.3, 4.3.1.2

[100] K.Nummiaro, E.Koller-Meier, L.Van Gool,"An adaptive color-based particle filter", *Image and Vision Computing*, Vol.2(1), pp.99-110, 2003. 3.2.2, 3.3, 4.3, 4.3.2.1, 4.4.2

[101] B.Parhami, "Voting algorithms", *IEEE Transactions on Reliability*, Vol.43(3), pp.617-629, 1994. 4.2

[102] P.Perez, C.Hue, J.Vermaak, M.Gangnet, "Color-based probabilistic Tracking", *Proc. European Conference on Computer Vision (ECCV)*, pp.661-675, 2002. 3.2.2

[103] F.Pighin, R.Szeliski, D.Salesin, "Resynthesizing facial animation through 3d model-based tracking", *Proc. International Conference on Computer Vision (ICCV)*, pp.143-150, 1999. 6.2

[104] P.Pirjanian, H.I.Christensen, J.A.Fayman, "Application of voting to fusion of purposive modules: An experimental investigation", *Robotics and Autonomous Systems*, Vol.23, pp.253-266, 1998. 4.3, 4.3.1.2

[105] P.Prokopowicz, R.Swain, R.Kahn, "Task and environment-sentitive tracking", *Proc. IEEE Symposium on Visual Languages*, pp.73-78, 1994. 4.3, 4.3.1.1, 4.4.2, 4.4.3

[106] R.Ramamoorthi, P.Hanrahan, "A signal-processing framework for inverse rendering", *Proc. of SIGGRAPH, ACM*, pp.117-128, 2001. 6.1, 6.2

[107] C.Rasmussen, G.Hager, "Joint probabilistic techniques for tracking multi-part objects", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.16-21, 1998. 2.2, 4.3, 4.3.2.3

[108] C.Rasmussen, G.D.Hager, "Probabilistic data association methods for tracking complex visual objects", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.23(6), pp.560-576, 2001. 3.2.2

[109] Y.Raja, S.McKenna, S.Gong, "Color model selection and adaption in dynamic scenes", *Proc. European Conference on Computer Vision (ECCV)*, Vol.1, pp.460-475, 1998. 3.1, 3.2.2, 3.3, 3.3.2

[110] Y.Schechner, S.Nayar, P.Belhumeur, "A theory of multiplexed illumination", *Proc. International Conference on Computer Vision (ICCV)*, Vol.2, pp.808-815, 2003. 6.1, 6.2, 6.6.1

[111] B.Scholkopf, A.Smola, K.R.Muller, "Kernel principal component analysis", *Advances in Kernel Methods - Suppor Vector Learning, MIT Press*, pp.327-352, 1999. 7.1.1

[112] K.Shearer, K.D.Wong, S.Venkatesh, "Combining multiple tracking algorithms for improved general performance", *Pattern Recognition*, Vol.34, pp.1257-1269, 2001. 4.3, 4.3.1.1, 4.4.2

[113] C.Shen, A.Hengel, A.Dick, "Probabilistic multiple cue integration for particle filter based tracking", *Proc. Digital Image Computing: Techniques and Applications*, pp.399-408, 2003. 4.3, 4.3.2.1, 4.4.2

[114] J.Sherrah, S.Gong, "Continuous global evidence-based bayesian modality fusion for simultaneous tracking of multiple objects", *Proc. International Conference on Computer Vision (ICCV)*, Vol.2, pp.42-49, 2001. 3.2.2, 4.3, 4.3.1.3, 4.4.2

[115] J.Sherrah, S.Gong, "Fusion of perceptual cues for robust tracking of head pose and position", *Pattern Recognition*, Vol.34(8), pp.1565-1572, 2001. 4.3, 4.3.2.1

[116] J.Shi, J.Malik, "Motion segmentation and tracking using normalized cuts", *Proc. International Conference on Computer Vision (ICCV)*, pp.1154-1160, 1998. 4.3, 4.3.2.2

[117] J.Shi, J.Malik, "Normalized cuts and image segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22(8), pp.888-905, 2000. 4.3, 4.3.2.2, 4.4.3

[118] N.T.Siebel, S.Maybank, "Fusion of multiple tracking algorithms for robust people tracking", *Proc. European Conference on Computer Vision (ECCV)*, Vol.4, pp.373-387, 2002. 4.3, 4.3.2.1, 4.4.3

[119] L.Sigal, S.Sclaroff, V.Athitsos, "Estimation and prediction of evolving color distributions for skin segmentation under varying illumination", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol.2, pp.152-159, 2000. 3.1, 3.2.2, 3.3

[120] H.Sidenbladh, F.De la Torre, M.J.Black, "A framework for modeling the appearance of 3D articulated figures", *Proc. IEEE International Conference on Automatic Face and Gesture Recognition (F&G)*, pp.368-375, 2000. 3.1

[121] W.Skarbek, A.Koschan, "Color segmentation survey", *Technical report 94-32, University of Berlin*, 1994. 3.2

[122] M.Spengler, B.Schiele, "Towards Robust Multi-Cue Integration for Visual Tracking", *Machine Vision and Applications*, Vol.14(1), pp. 50-58, 2003. 3.2.2, 4.3, 4.3.2.1

[123] M.J.Swain, D.H.Ballard, "Color indexing", *International Journal of Computer Vision*, Vol.7(1), pp.11-32, 1991. 3.1, 3.3

[124] G.Taylor, L.Kleeman, "Fusion of multimodal visual cues for model-based object tracking", *Proc. Australasian Conference on Robotics and Automation*, 2003. 3.2.2, 4.3, 4.3.1.3

[125] H.Tao, H.Sawhney, R.Kumar, "Dynamic layer representation with applications to tracking", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol.2, pp.134-141, 2000. 4.3, 4.3.2.2

[126] J.C.Terrillon, S.Akamatsu, "Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images", *Proc. Vision Interface*, pp.180-187, 1999. 3.2.1

[127] P.H.S.Torr, "Geometric motion segmentation and model selection", *Philosophical Transactions of the Royal Society A*, pp.1321-1340, 1998. 3.1

[128] P.H.S.Torr, A.Zisserman, "Concerning bayesian motion segmentation, model averaging, mathching and the trifocal tensor", *Proc. European Conference on Computer Vision (ECCV)*, Vol.1, pp.511-528, 1998. 3.1

[129] K.Toyama, G.Hager, "Incremental focus of attention for robust vision tracking", *International Journal of Computer Vision*, Vol.35(1), pp.45-63, 1999. 4.3, 4.3.2.1

[130] K.Toyama, E.Horvitz, "Bayesian modality fusion: Probabilistic integration of multiple vision algorithms for head tracking", *Proc. Asian Conference on Computer Vision (ACCV)*, 2000. 4.3, 4.3.1.3, 4.4.2

[131] J.Triesch, C.von der Malsburg, "Democratic Integration: Self-Organized Integration of Adaptive Cues", *Neural Computation*, Vol.13(9), pp.2049-2074, 2001. 3.2.2, 4.3, 4.3.1.1, 4.3.1.3, 4.3.2.1, 4.4.1, 4.4.2, 4.4.3

[132] T.Vetter, V.Blanz, "Coloured 3D face models from single images: An example based approach", *Proc. European Conference on Computer Vision (ECCV)*, pp.499-513, 1998. 4.3, 4.3.2.2

[133] C.Wallace, D.Dowe, "Minimum message lenght and Kolmogorov complexity", *The Computer Journal*, Vol.42(4), pp.270-283, 1999. 3.3.2

[134] E.A.Wan, R.V.der Merwe, "The unscented Kalman filter for nonlinear estimation", *Proc. Symposium on Adaptive Systems for Signal Processing, Communications and Control*, 2000. 2.2

[135] Y.Wang, X.Huang, C.Lee, S.Zhang, Z.Li, D.Samaras, D.Metaxas, A.Elgammal, P.Huang, "High resolution acquisition, learning and transfer of dynamic 3D facial", *Proc. Computer Graphics Forum*, Vol.23(3), pp.677-686, 2004 6.1.2, 6.6.4

[136] G.Welch, G.Bishop, "An introduction to the Kalman filter. Course 8", *Proc. of SIGGRAPH, ACM*, August 2001. 2.2, A

[137] T.Wong, C.Fu, P.Heng, C.Leung, "The plenoptic illumination function", *IEEE Transactions on Multimedia*, Vol.4(3), pp.361-371, 2002. 6.3.1

[138] T.Wong, P.Heng, S.Or, Y.Ng, "Image-based rendering with controllable illumination", *Proc. Eurographics Symposium on Rendering*, pp.13-22, 1997. 6.1

[139] C.Wren, A.Azarbayejani, T.Darrel, A.Pentland, "Pfinder:Real-time tracking of the human body", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19(7), pp.780-785, 1997. 3.2.2, 4.3, 4.3.2.2

[140] Y.Wu, T.S.Huang, "A co-inference approach to robust visual tracking", *Proc. International Conference on Computer Vision (ICCV)*, Vol.2, pp.26-33, 2001. 3.2.2

[141] Y.Wu, T.S.Huang, "Nonstationary color tracking for vision-based human-computer interaction", *IEEE Transactions on Neural Networks*, Vol.13(4), pp.948-960, 2002. 3.1, 3.2.2

[142] Y.Wu, T.S.Huang, "Robust visual tracking by integrating multiple cues based on co-inference learning", *International Journal of Computer Vision*, Vol.58(1), pp.55-71,2004. 4.3, 4.3.2.3, 4.4.2, 5.1, 5.3.1, 7.2

[143] C.Xu, J.L.Prince, "Snakes, shapes, and gradient vector flow", *IEEE Transactions on Image Processing*, Vol.7(3), pp.359-369, 1998. 3.4.1, 3.12

[144] M.H.Yang, N.Ahuja, "Detecting human faces in color images", *Proc. IEEE International Conference on Image Processing (ICIP)*, Vol.1, pp.127-130, 1998. 3.1, 3.3, 3.3.2

[145] J.Yang, W.Lu, A.Waibel, "Skin-color modeling and adaption", *Proc. Asian Conference on Computer Vision (ACCV)*, Vol.2, pp.687-694, 1998. 3.2.2, 3.3

[146] L.A.Zadeh, "Fuzzy sets and applications: Selected papers by L.A.Zadeh", *New York: John Wiley & Sons*, 1987. 4.2

[147] L.Zhang, B.Curless, S.Seitz, "Spacetime stereo: shape recovery for dynamic scenes ", *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.367-374, 2003. 6.1.2, 6.6.4

Dentro de mi corazón
yo conservo una ilusión
y por decir la verdad
no tengo fe en lo que espero,
mas tampoco desespero,
de que se haga realidad.

Ramon Noguer Fàbrega